# Basler racer

## USER'S MANUAL FOR GigE VISION CAMERAS

Document Number: AW001183
Version: 01 Language: 000 (English)
Release Date: 20 June 2012

**Preliminary**

**The information in this document is preliminary and all content is subject to change.**

**Applies to prototype cameras only.**

BASLER
the power of sight

**For customers in the U.S.A.**

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

You are cautioned that any changes or modifications not expressly approved in this manual could void your authority to operate this equipment.

The shielded interface cable recommended in this manual must be used with this equipment in order to comply with the limits for a computing device pursuant to Subpart J of Part 15 of FCC Rules.

**For customers in Canada**

This apparatus complies with the Class A limits for radio noise emissions set out in Radio Interference Regulations.

**Pour utilisateurs au Canada**

Cet appareil est conforme aux normes Classe A pour bruits radioélectriques, spécifiées dans le Règlement sur le brouillage radioélectrique.

**Life Support Applications**

These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Basler customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Basler for any damages resulting from such improper use or sale.

**Warranty Note**

Do not open the housing of the camera. The warranty becomes void if the housing is opened.

**All material in this publication is subject to change without notice and is copyright Basler AG.**

## Contacting Basler Support Worldwide

**Europe:**

Basler AG
An der Strusbek 60 - 62
22926 Ahrensburg
Germany

Tel.: +49-4102-463-515
Fax.: +49-4102-463-599

bc.support.europe@baslerweb.com

**Americas:**

Basler, Inc.
855 Springdale Drive, Suite 203
Exton, PA 19341
U.S.A.

Tel.: +1-610-280-0171
Fax.: +1-610-280-7608

bc.support.usa@baslerweb.com

**Asia:**

Basler Asia Pte. Ltd.
8 Boon Lay Way
# 03 - 03 Tradehub 21
Singapore 609964

Tel.: +65-6425-0472
Fax.: +65-6425-0473

bc.support.asia@baslerweb.com

**www.baslerweb.com**

# Table of Contents

**1 Specifications, Requirements, and Precautions . . . . . 1**
1.1 Models . . . . . 1
1.2 General Specifications . . . . . 2
1.3 Spectral Response . . . . . 3
1.4 Mechanical Specifications . . . . . 4
1.4.1 Camera Dimensions and Mounting Points . . . . . 4
1.4.2 Lens Adapter Dimensions . . . . . 6
1.5 Software Licensing Information . . . . . 7
1.6 Avoiding EMI and ESD Problems . . . . . 8
1.7 Environmental Requirements . . . . . 9
1.7.1 Temperature and Humidity . . . . . 9
1.7.2 Heat Dissipation . . . . . 9
1.8 Precautions . . . . . 10

**2 Software and Hardware Installation . . . . . 13**

**3 Tools for Changing Camera Parameters . . . . . 15**
3.1 The pylon Viewer . . . . . 15
3.2 The IP Configuration Tool . . . . . 15
3.3 The pylon API . . . . . 16

**4 Basler Network Drivers and Parameters . . . . . 17**
4.1 The Basler Filter Driver . . . . . 18
4.2 The Basler Performance Driver . . . . . 19
4.3 Transport Layer Parameters . . . . . 27

**5 Network Related Camera Parameters and Managing Bandwidth . . . . . 29**
5.1 Network Related Parameters in the Camera . . . . . 29
5.2 Managing Bandwidth When Multiple Cameras Share a Single Network Path . . . . . 31
5.2.1 A Procedure for Managing Bandwidth . . . . . 32

**6 Camera Functional Description . . . . . 37**

**7 Physical Interface . . . . . 39**
7.1 General Description of the Connections . . . . . 39
7.1.1 Pin Numbering . . . . . 40
7.2 Connector Pin Assignments . . . . . 41
7.2.1 Pin Assignments for the 6-pin Receptacle . . . . . 41
7.2.2 Pin Assignments for the 12-pin Receptacle . . . . . 42
7.2.3 Pin Assignments for the RJ-45 Jack . . . . . 43

7.3 Connector Types . . . . . 44
7.3.1 6-pin Connector . . . . . 44
7.3.2 12-pin Connector . . . . . 44
7.3.3 RJ-45 Jack . . . . . 44
7.4 Cabling Requirements . . . . . 45
7.4.1 Power Cable . . . . . 45
7.4.2 I/O Cable . . . . . 46
7.4.3 Ethernet Cables . . . . . 46
7.5 Camera Power . . . . . 47
7.6 Input and Output Lines . . . . . 48
7.6.1 Input Lines . . . . . 48
7.6.1.1 Electrical Characteristics . . . . . 48
7.6.1.2 Input Line Debouncers . . . . . 52
7.6.1.3 Input Line Inverters . . . . . 53
7.6.1.4 Selecting an Input Line as a Source Signal for a Camera Function . . . . . 53
7.6.2 Output Lines . . . . . 55
7.6.2.1 Electrical Characteristics . . . . . 55
7.6.2.2 Minimum Output Pulse Width . . . . . 58
7.6.2.3 Output Line Inverters . . . . . 58
7.6.2.4 Selecting the Source Signal for an Output Line . . . . . 59
7.6.2.5 Setting the State of User Settable Output Lines . . . . . 60
7.6.3 Checking the State of the I/O Lines . . . . . 62
7.6.4 Checking the Line Logic . . . . . 63
7.6.5 I/O Line Response Times . . . . . 63
7.7 Ethernet GigE Device Information . . . . . 64

**8 Acquisition Control . . . . . 65**

8.1 Defining a Frame . . . . . 65
8.2 Controlling Acquisition . . . . . 68
8.2.1 Acquisition Start and Stop Commands and the Acquisition Mode . . . . . 68
8.2.2 Acquisition Start Triggering . . . . . 69
8.2.2.1 Acquisition Start Trigger Mode = Off . . . . . 70
8.2.2.2 Acquisition Start Trigger Mode = On . . . . . 70
8.2.2.3 Acquisition Frame Count . . . . . 71
8.2.2.4 Setting The Acquisition Start Trigger Mode and Related Parameters . . . . . 71
8.2.3 Frame Start Triggering . . . . . 72
8.2.3.1 Frame Start Trigger Mode = Off . . . . . 72
8.2.3.2 Frame Start Trigger Mode = On . . . . . 73
8.2.3.3 Setting the Frame Start Trigger Parameters . . . . . 75
8.2.3.4 Frame Timeout . . . . . 76

8.2.4 Line Start Triggering . . . 77
8.2.4.1 Line Start Trigger Mode = Off . . . 77
8.2.4.2 Line Start Trigger Mode = On . . . 78
8.2.4.3 Setting the Line Start Trigger Parameters . . . 82
8.2.5 Exposure Time . . . 83
8.2.5.1 Minimum and Maximum Exposure Times . . . 83
8.2.5.2 Exposure Time Parameters . . . 84
8.2.6 Use Case Descriptions and Diagrams . . . 86
8.3 The Shaft Encoder Module . . . 104
8.4 Frequency Converter . . . 112
8.5 Acquisition Monitoring Tools . . . 114
8.5.1 Exposure Active Signal . . . 114
8.5.2 Acquisition Status . . . 115
8.5.3 Acquisition Trigger Wait Signal . . . 116
8.5.4 Frame Trigger Wait Signal . . . 116
8.5.5 Line Trigger Wait Signal . . . 117
8.5.6 Input Related Signals as Output Signals . . . 117
8.6 Frame Transmission Time . . . 118
8.7 Maximum Allowed Line Acquisition Rate . . . 119

**9 Pixel Data Formats . . . 123**
9.1 Setting the Pixel Data Format . . . 123
9.2 Pixel Data Formats . . . 124
9.2.1 Mono 8 Format . . . 124
9.2.2 Mono 12 Format . . . 125
9.2.3 Mono 12 Packed Format . . . 127
9.2.4 YUV 4:2:2 Packed Format . . . 129
9.2.5 YUV 4:2:2 (YUYV Packed) Format . . . 131
9.3 Pixel Transmission Sequence . . . 133

**10 Standard Features . . . 135**
10.1 Gain and Black Level . . . 135
10.1.1 Gain . . . 135
10.1.1.1 Analog Gain . . . 136
10.1.1.2 Digital Gain . . . 136
10.1.1.3 Using Both Analog Gain and Digital Gain . . . 138
10.1.2 Black Level . . . 138
10.2 Remove Parameter Limits . . . 140
10.3 Event Reporting . . . 141
10.4 Luminance Lookup Table . . . 144
10.5 Binning . . . 147
10.6 Gamma Correction . . . 148

10.7 Shading Correction . . . . . 149
10.7.1 Offset Shading Correction . . . . . 149
10.7.2 Gain Shading Correction . . . . . 149
10.7.3 Default Shading Set File and User Shading Set File . . . . . 150
10.7.3.1 Creating a "Usershading" File . . . . . 150
10.7.3.2 Working with Shading Sets . . . . . 153
10.8 Trigger Delay . . . . . 154
10.9 Error Codes . . . . . 155
10.10 Test Images . . . . . 157
10.10.1 Test Images in Detail . . . . . 158
10.11 Device Information Parameters . . . . . 160
10.12 User Defined Values . . . . . 162
10.13 Configuration Sets . . . . . 163
10.13.1 Saving Configuration Sets . . . . . 164
10.13.2 Loading a Saved Set or the Default Set into the Active Set . . . . . 165
10.13.3 Selecting the Default Startup Set . . . . . 165

**11 Chunk Features . . . . . 167**
11.1 What are Chunk Features? . . . . . 167
11.2 Making the "Chunk Mode" Active and Enabling the Extended Data Stamp . . . . . 168
11.3 Frame Counter . . . . . 170
11.4 Time Stamp . . . . . 173
11.5 Trigger Counters . . . . . 175
11.6 Encoder Counter . . . . . 178
11.7 Input Line Status At Line Trigger . . . . . 180
11.8 CRC Checksum . . . . . 182

**12 Troubleshooting and Support . . . . . 185**
12.1 Tech Support Resources . . . . . 185
12.2 Obtaining an RMA Number . . . . . 185
12.3 Before Contacting Basler Technical Support . . . . . 186

**Revision History** . . . . . 189

**Index** . . . . . 191

# 1 Specifications, Requirements, and Precautions

This chapter lists the camera models covered by the manual. It provides the general specifications for those models and the basic requirements for using them.

This chapter also includes specific precautions that you should keep in mind when using the cameras. We strongly recommend that you read and follow the precautions.

## 1.1 Models

The current Basler racer GigE Vision camera models are listed in the top row of the specification tables on the next pages of this manual. The camera models are differentiated by their sensor size and their maximum line rate at full resolution.

Unless otherwise noted, the material in this manual applies to all of the camera models listed in the tables. Material that only applies to a particular camera model or to a subset of models will be so designated.

# 1.2 General Specifications

| Specification | raL2048-48gm | raL4096-24gm |
|---|---|---|
| Sensor Size | 2048 pixels | 4096 pixels |
| Sensor Type | Awaiba DR-2k-7<br>Monochrome Linear CMOS | Awaiba DR-4k-7<br>Monochrome Linear CMOS |
| Pixel Size | 7 µm x 7 µm | |
| Max Line Rate | 48 kHz | 24 kHz |
| Min Line Rate | No minimum when an external line trigger signal is used<br>100 Hz when an external line trigger signal is not used | |
| Mono/Color | Mono | |
| Data Output Type | Fast Ethernet (100 Mbit/s) or Gigabit Ethernet (1000 Mbit/s) | |
| Pixel Data Formats | Mono 8<br>Mono 12<br>Mono 12 Packed<br>YUV 4:2:2 Packed<br>YUV 4:2:2 (YUYV) Packed | |
| ADC Bit Depth | 12 bits | |
| Synchronization | Via external trigger signal, via software or free run | |
| Exposure Control | Programmable via the camera API | |
| Power Requirements | +12 VDC (± 10 %), < 1 % ripple | |
| Max Power Consumption (at 12 VDC) | < 4.5 W | < 5 W |
| I/O Lines | 3 input lines and 2 output lines | |
| Lens Adapter | Universal camera front, suitable for C-mount (available as accessory) | |
| Size (L x W x H) | 36.12 mm x 56 mm x 62 mm (without lens adapter or connectors)<br>50.92 mm x 56 mm x 62 mm (with C-mount lens adapter and connectors) | |
| Weight | ~ 250 g (typical) without lens adapter<br>~ 280 g (typical) with C-mount lens adapter | |
| Conformity | CE, FCC, KCC, UL (in preparation), GenICam, GigE Vision, IP30 | |

Table 1: General Specifications - 2k and 4k Mono Cameras

# 1.3 Spectral Response

The following graph shows the quantum efficiency curve for monochrome cameras.

| | |
|---|---|
| | The quantum efficiency curve excludes lens characteristics and light source characteristics. |

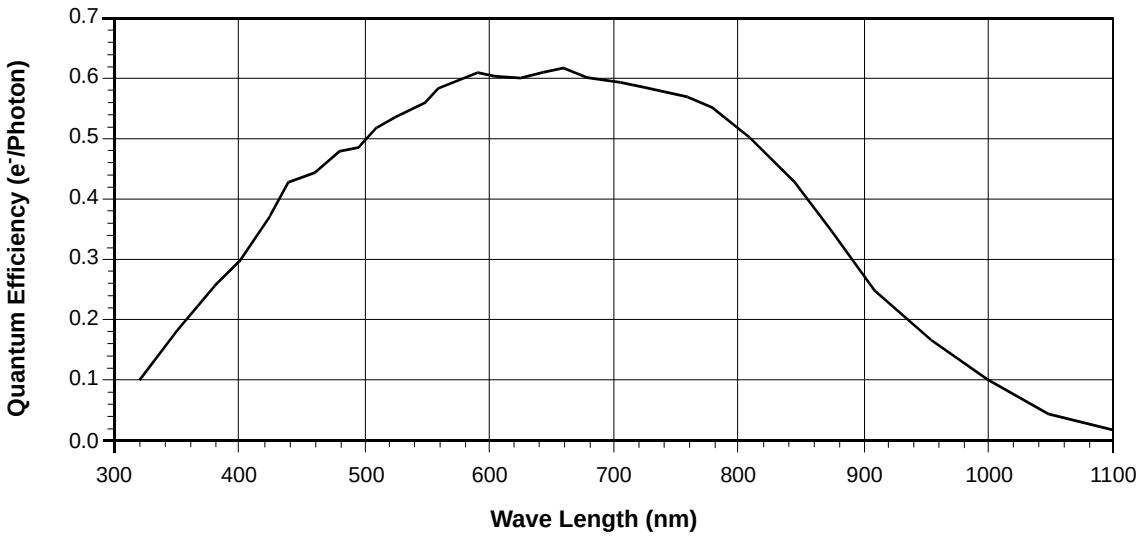



Fig. 1: Monochrome Camera Quantum Efficiency (From Sensor Data Sheet)

# 1.4 Mechanical Specifications

## 1.4.1 Camera Dimensions and Mounting Points

The cameras are manufactured with high precision. Planar, parallel, and angular sides guarantee precise mounting with high repeatability.

The camera housings conform to the IP30 protection class provided the camera front or the lens mount is covered by the protective plastic seal that is shipped with the camera.

The camera’s dimensions in millimeters are as shown in the drawings below.

Camera housings are equipped with four mounting holes on the front and two mounting holes on each side as shown in the drawings.

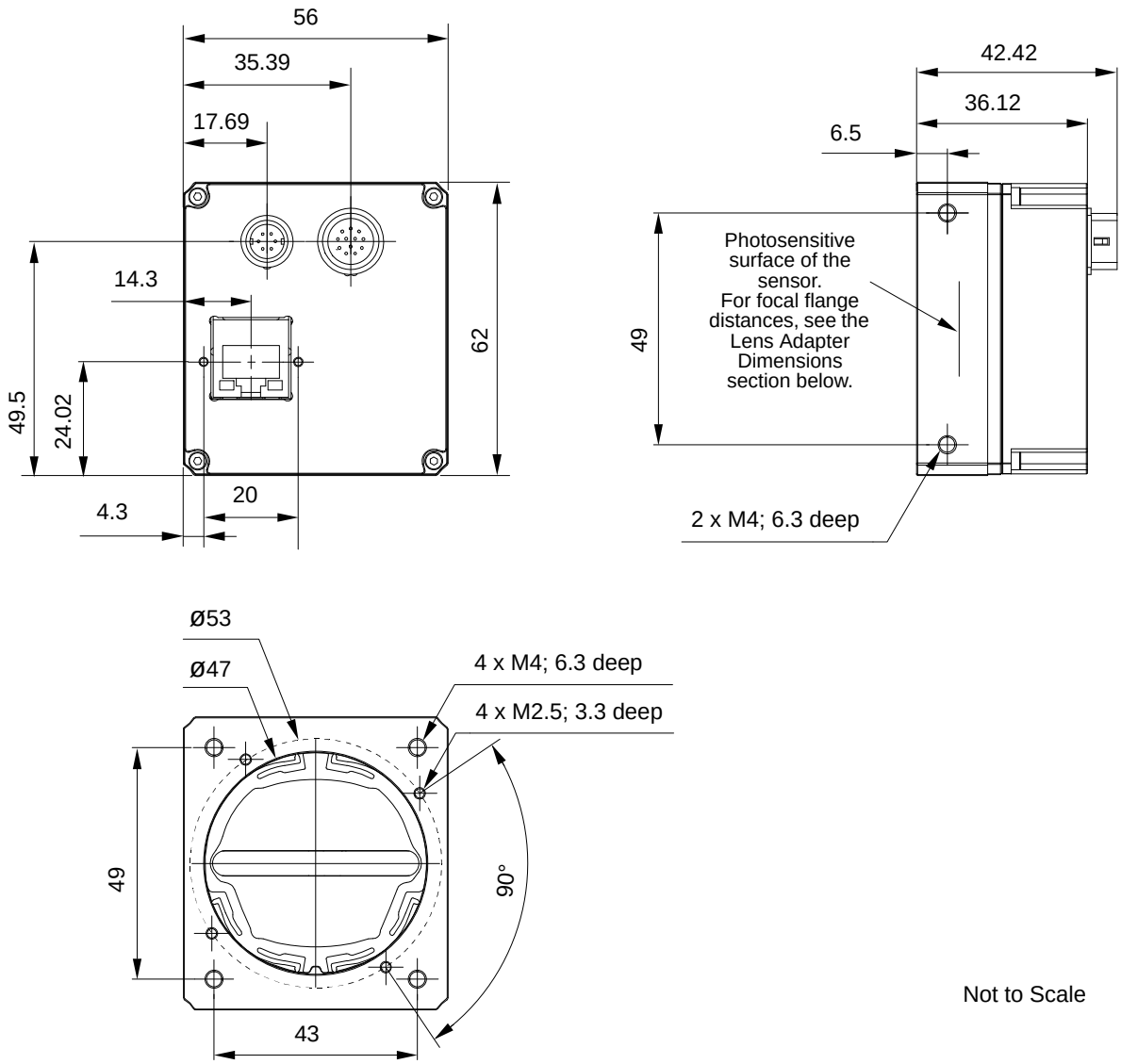



Fig. 2: Mechanical Dimensions (in mm)

# 1.4.2 Lens Adapter Dimensions

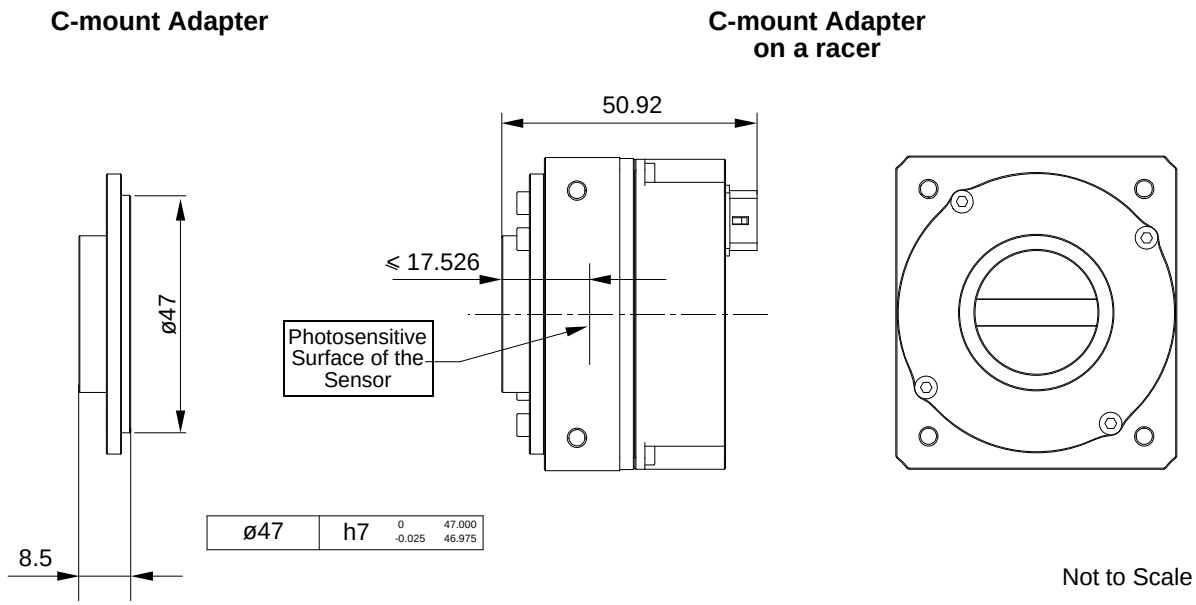



Fig. 3: C-mount Adapter Dimensions

# 1.5 Software Licensing Information

The software in the camera includes the LWIP TCP/IP implementation. The copyright information for this implementation is as follows:

Copyright (c) 2001, 2002 Swedish Institute of Computer Science. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.

IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# 1.6 Avoiding EMI and ESD Problems

The cameras are frequently installed in industrial environments. These environments often include devices that generate electromagnetic interference (EMI) and they are prone to electrostatic discharge (ESD). Excessive EMI and ESD can cause problems with your camera such as false triggering or can cause the camera to suddenly stop capturing images. EMI and ESD can also have a negative impact on the quality of the image data transmitted by the camera.

To avoid problems with EMI and ESD, you should follow these general guidelines:

- Always use high quality shielded cables. The use of high quality cables is one of the best defenses against EMI and ESD.
- Try to use camera cables that are as short as possible and try to run the camera cables and power cables parallel to each other. Avoid coiling camera cables. If the cables are too long, use a meandering path rather then coiling the cables.
- Avoid placing camera cables parallel to wires carrying high-current, switching voltages such as wires supplying stepper motors or electrical devices that employ switching technology. **Placing camera cables near to these types of devices may cause problems with the camera.**
- Attempt to connect all grounds to a single point, e.g., use a single power outlet for the entire system and connect all grounds to the single outlet. This will help to avoid large ground loops. (Large ground loops can be a primary cause of EMI problems.)
- Use a line filter on the main power supply.
- Install the camera and camera cables as far as possible from devices generating sparks. If necessary, use additional shielding.
- Decrease the risk of electrostatic discharge by taking the following measures:
  - Use conductive materials at the point of installation (e.g., floor, workplace).
  - Use suitable clothing (cotton) and shoes.
  - Control the humidity in your environment. Low humidity can cause ESD problems.

The Basler application note called *Avoiding EMI and ESD in Basler Camera Installations* provides much more detail about avoiding EMI and ESD.
This application note can be obtained from the Downloads section of our website:
www.baslerweb.com

# 1.7 Environmental Requirements

## 1.7.1 Temperature and Humidity

| | |
|---|---|
| Housing temperature during operation: | 0 °C ... +50 °C (+32 °F ... +122 °F) |
| Humidity during operation: | 20 % ... 80 %, relative, non-condensing |
| Storage temperature: | -20 °C ... +80 °C (-4 °F ... +176 °F) |
| Storage humidity: | 20 % ... 80 %, relative, non-condensing |

## 1.7.2 Heat Dissipation

You must provide sufficient heat dissipation to maintain the temperature of the camera housing at 50 °C or less. Since each installation is unique, Basler does not supply a strictly required technique for proper heat dissipation. Instead, we provide the following general guidelines:

- In all cases, you should monitor the temperature of the camera housing and make sure that the temperature does not exceed 50 °C. Keep in mind that the camera will gradually become warmer during the first hour of operation. After one hour, the housing temperature will have stabilized and will no longer increase.
- If your camera is mounted on a substantial metal component in your system, this may provide sufficient heat dissipation.
- The use of a fan to provide air flow over the camera is an extremely efficient method of heat dissipation. The use of a fan provides the best heat dissipation.

# 1.8 Precautions

**NOTICE**

Avoid dust on the sensor.

The camera is shipped with a protective plastic seal on the camera front or lens mount. To avoid collecting dust on the camera's sensor, make sure that you always put the protective seal in place when there is no lens mounted on the camera.

Also, make sure to always point the camera downward when there is no protective seal or lens on the camera front or lens mount.

**NOTICE**

- Applying incorrect power can damage the camera.

  The camera's required nominal operating voltage is +12 VDC (±10 %), effective on the camera's connector.
- Applying power with the wrong polarity can severely damage the camera.

  Make sure that the polarity of the power applied to the camera is correct. Applying power with the wrong polarity can severely damage the camera.

**NOTICE**

- Using a wrong pin assignment for the 12-pin receptacle can severely damage the camera.

  Make sure the cable and plug you connect to the 12-pin receptacle follows the correct pin assignment. In particular, do **not** use a pin assignment that would be correct for Basler area scan cameras. The 12-pin receptacles of Basler line scan and area scan cameras are electrically incompatible.

**NOTICE**

Incorrect plugs can damage the camera's connectors.

The plug on the cable that you attach to the camera's 6-pin connector must have 6 female pins. The plug on the cable that you attach to the camera's 12-pin connector must have 12 female pins.

**NOTICE**

Inappropriate code may cause unexpected camera behavior.

1. The code snippets provided in this manual are included as sample code only. Inappropriate code may cause your camera to function differently than expected and may compromise your application.
2. To ensure that the snippets will work properly in your application, you must adjust them to meet your specific needs and must test them thoroughly prior to use.
3. The code snippets in this manual are written in C++. Other programming languages can also be used to write code for use with Basler pylon. When writing code, you should use a programming language that is both compatible with pylon and appropriate for your application. For more information about the programming languages that can be used with Basler pylon, see the documentation included with the pylon package.

## Warranty Precautions

**To ensure that your warranty remains in force:**

### Do not remove the camera's serial number label

If the label is removed and the serial number can't be read from the camera's registers, the warranty is void.

### Do not open the camera housing

Do not open the housing. Touching internal components may damage them.

### Keep foreign matter outside of the camera

Be careful not to allow liquid, flammable, or metallic material inside of the camera housing. If operated with any foreign matter inside, the camera may fail or cause a fire.

### Avoid Electromagnetic fields

Do not operate the camera in the vicinity of strong electromagnetic fields. Avoid electrostatic charging.

### Transport Properly

Transport the camera in its original packaging only. Do not discard the packaging.

### Clean Properly

Avoid cleaning the surface of the camera's sensor if possible. If you must clean it, use a soft, lint free cloth dampened with a small quantity of high quality window cleaner. Because electrostatic

discharge can damage the sensor, you must use a cloth that will not generate static during cleaning (cotton is a good choice).

To clean the surface of the camera housing, use a soft, dry cloth. To remove severe stains, use a soft cloth dampened with a small quantity of neutral detergent, then wipe dry.

Do not use solvents or thinners to clean the housing; they can damage the surface finish.

### Read the manual

Read the manual carefully before using the camera!

# 2 Software and Hardware Installation

The information you will need to install and operate the camera is included in the Installation and Setup Guide for Cameras Used with Basler’s pylon API (AW000611xx000).

You can download the Installation and Setup Guide for Cameras Used with Basler’s pylon API from the Basler website: www.baslerweb.com

The guide includes the information you will need to install both hardware and software and to begin capturing images. It also describes the recommended network adapters, describes the recommended architecture for the network to which your camera is attached, and deals with the IP configuration of your camera and network adapter.

After completing your camera installation, refer to the "Basler Network Drivers and Parameters" and "Network Related Camera Parameters and Managing Bandwidth" sections of this camera User’s Manual for information about improving your camera’s performance in a network and about using multiple cameras.

# 3 Tools for Changing Camera Parameters

This chapter explains the options available for changing the camera's parameters. The available options let you change parameters either by using stand-alone tools that access the camera via a GUI or by accessing the camera from within your software application.

## 3.1 The pylon Viewer

The Basler pylon Viewer is a standalone application that lets you view and change most of the camera's parameter settings via a GUI based interface. The viewer also lets you acquire images, display them, and save them. Using the pylon Viewer software is a very convenient way to get your camera up and running quickly when you are doing your initial camera evaluation or doing a camera design-in for a new project.

The pylon Viewer is included in Basler's pylon Driver Package. You can obtain the pylon package from the Downloads section of our website: www.baslerweb.com

For more information about using the viewer, see the installation and Setup Guide for Cameras Used with Basler's pylon API (AW000611xx000). You can download the guide from the Basler website: www.baslerweb.com.

## 3.2 The IP Configuration Tool

The Basler IP Configuration Tool is a standalone application that lets you change the IP configuration of the camera via a GUI. The tool will detect all Basler GigE cameras attached to your network and let you make changes to a selected camera.

The IP Configuration Tool is included in Basler's pylon Driver Package. You can obtain the pylon package from the Downloads section of our website: www.baslerweb.com

For more information about using IP Configuration Tool, see the installation and Setup Guide for Cameras Used with Basler's pylon API (AW000611xx000). You can download the guide from the Basler website: www.baslerweb.com.

# 3.3 The pylon API

You can access all of the camera's parameters and can control the camera's full functionality from within your application software by using Basler's pylon API. The Basler pylon Programmer's Guide and API Reference contains an introduction to the API and includes information about all of the methods and objects included in the API. The programmer's guide and API reference are included in the pylon SDK.

The Basler pylon Software Development Kit (SDK) includes a set of sample programs that illustrate how to use the pylon API to parameterize and operate the camera. These samples include Microsoft® Visual Studio® solution and project files demonstrating how to set up the build environment to build applications based on the API.

The SDK is available in the Downloads section of the Basler website: www.baslerweb.com

For more information about installing pylon software, see the installation and Setup Guide for Cameras Used with Basler's pylon API (AW000611xx000). You can download the guide from the Basler website: www.baslerweb.com.

# 4 Basler Network Drivers and Parameters

This section describes the Basler network drivers available for your camera and provides detailed information about the parameters associated with the drivers.

Two network drivers are available for the network adapter used with your GigE cameras:

- The **Basler filter driver** is a basic GigE Vision network driver that is compatible with all network adapters. The advantage of this driver is its extensive compatibility.
- The **Basler performance driver** is a hardware specific GigE Vision network driver. The driver is only compatible with network adapters that use specific Intel chipsets. The advantage of the performance driver is that it significantly lowers the CPU load needed to service the network traffic between the PC and the camera(s). It also has a more robust packet resend mechanism.

During the installation process you should have installed either the filter driver or the performance driver.

For more information about compatible Intel chipsets and about installing the network drivers, see the Installation and Setup Guide for Cameras Used with Basler’s pylon API.

# 4.1 The Basler Filter Driver

The Basler filter driver is a basic driver GigE Vision network driver. It is designed to be compatible with most network adapter cards.

The functionality of the filter driver is relatively simple. For each frame, the driver checks the order of the incoming packets. If the driver detects that a packet or a group of packets is missing, it will wait for a specified period of time to see if the missing packet or group of packets arrives. If the packet or group does not arrive within the specified period, the driver will send a resend request for the missing packet or group of packets.

The parameters associated with the filter driver are described below.

**Enable Resend** - Enables or disables the packet resend mechanism.

If packet resend is disabled and the filter driver detects that a packet has been lost during transmission, the grab result for the returned buffer holding the image will indicate that the grab failed and the image will be incomplete.

If packet resend is enabled and the driver detects that a packet has been lost during transmission, the driver will send a resend request to the camera. If the camera still has the packet in its buffer, it will resend the packet. If there are several lost packets in a row, the resend requests will be combined.

**Packet Timeout** - The Packet Timeout parameter defines how long (in milliseconds) the filter driver will wait for the next expected packet before it initiates a resend request. Ensure the Packet Timeout parameter is set to a longer time interval than the time interval set for the inter-packet delay.

**Frame Retention** - The Frame Retention parameter sets the timeout (in milliseconds) for the frame retention timer. Whenever the filter driver detects the leader for a frame, the frame retention timer starts. The timer resets after each packet in the frame is received and will timeout after the last packet is received. If the timer times out at any time before the last packet is received, the buffer for the frame will be released and will be indicated as an unsuccessful grab.

You can set the filer driver parameter values from within your application software by using the Basler pylon API. The following code snippet illustrates using the API to read and write the parameter values:

```
// Enable Resend
Camera_t::StreamGrabber_t StreamGrabber ( Camera.GetStreamGrabber(0) );
StreamGrabber.EnableResend.SetValue(false); // disable resends

// Packet Timeout/FrameRetention
Camera_t::StreamGrabber_t StreamGrabber ( Camera.GetStreamGrabber(0) );
StreamGrabber.PacketTimeout.SetValue( 40 );
StreamGrabber.FrameRetention.SetValue( 200 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

# 4.2 The Basler Performance Driver

The Basler performance driver is a hardware specific GigE Vision network driver compatible with network adapters that use specific Intel chipsets. The main advantage of the performance driver is that it significantly lowers the CPU load needed to service the network traffic between the PC and the camera(s). It also has a more robust packet resend mechanism.

For more information about compatible Intel chipsets, see the installation and Setup Guide for Cameras Used with Basler's pylon API.

The performance driver uses two distinct "resend mechanisms" to trigger resend requests for missing packets:

- The threshold resend mechanism
- The timeout resend mechanism

The mechanisms are independent from each other and can be used separately. However, for maximum efficiency and for ensuring that resend requests will be sent for all missing packets, we recommend using both resend mechanisms in a specific, optimized combination, as provided by the parameter default values.

The performance driver's parameter values determine how the resend mechanisms act and how they relate to each other. You can set the parameter values by using the pylon Viewer or from within your application software by using the pylon API.

The parameter default values will provide for the following:

- The threshold resend mechanism precedes the timeout resend mechanism. This ensures that a resend request is sent for every missing packet, even at very high rates of arriving packets.
- The timeout resend mechanism will be effective for those missing packets that were not resent after the first resend request.

**We strongly recommend using the default parameter settings.** Only users with the necessary expertise should change the default parameter values.

The Basler performance driver uses a "receive window" to check the status of packets. The check for missing packets is made as packets enter the receive window. If a packet arrives from higher in the sequence of packets than expected, the preceding skipped packet or packets are detected as missing. For example, suppose packet (n-1) has entered the receive window and is immediately followed by packet (n+1). In this case, as soon as packet (n+1) enters the receive window, packet n will be detected as missing.

## General Parameters

**Enable Resend** - Enables the packet resend mechanisms.

If the Enable Resend parameter is set to false, the resend mechanisms are disabled. The performance driver will not check for missing packets and will not send resend requests to the camera.

If the Enable Resend parameter is set to true, the resend mechanisms are enabled. The performance driver will check for missing packets. Depending on the parameter settings and the resend response, the driver will send one or several resend requests to the camera.

**Receive Window Size** - Sets the size of the receive window.

## Threshold Resend Mechanism Parameters

The threshold resend request mechanism is illustrated in Fig. 4 where the following assumptions are made:

- Packets 997, 998, and 999 are missing from the stream of packets.
- Packet 1002 is missing from the stream of packets.

DIAGRAM IS NOT DRAWN TO SCALE

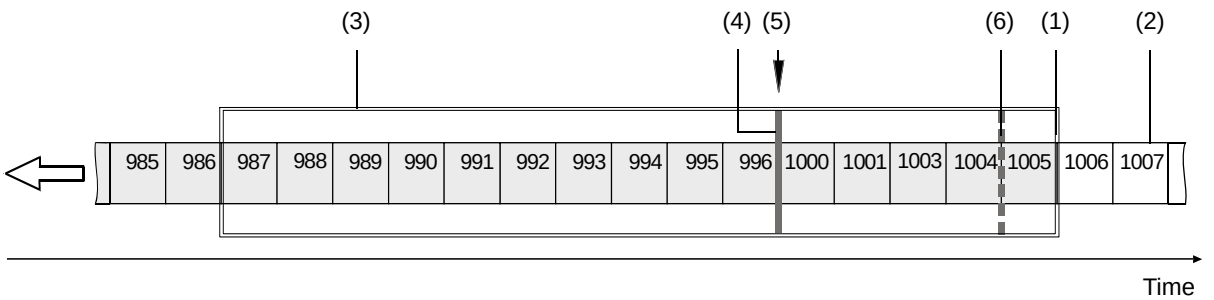



Fig. 4: Example of a Receive Window with Resend Request Threshold & Resend Request Batching Threshold

(1) Front end of the receive window. Missing packets are detected here.

(2) Stream of packets. Gray indicates that the status was checked as the packet entered the receive window. White indicates that the status has not yet been checked.

(3) Receive window of the performance driver.

(4) Threshold for sending resend requests (resend request threshold).

(5) A separate resend request is sent for each packets 997, 998, and 999.

(6) Threshold for batching resend requests for consecutive missing packets (resend request batching threshold). Only one resend request will be sent for the consecutive missing packets.

**Resend Request Threshold** - This parameter determines the location of the resend request threshold within the receive window as shown in Fig. 4. The parameter value is in per cent of the width of the receive window. In Fig. 4 the resend request threshold is set at 33.33% of the width of the receive window.

A stream of packets advances packet by packet beyond the resend request threshold (i.e. to the left of the resend request threshold in Fig. 4). As soon as the position where a packet is missing advances beyond the resend request threshold, a resend request is sent for the missing packet.

In the example shown in Fig. 4, packets 987 to 1005 are within the receive window and packets 997 to 999 and 1002 were detected as missing. In the situation shown, a resend request is sent to the camera for each of the missing consecutive packets 997 to 999. The resend requests are sent after packet 996 - the last packet of the intact sequence of packets - has advanced beyond the resend request threshold and before packet 1000 - the next packet in the stream of packets - can advance beyond the resend request threshold. Similarly, a resend request will be sent for missing packet 1002 after packet 1001 has advanced beyond the resend request threshold and before packet 1003 can advance beyond the resend request threshold.

**Resend Request Batching** - This parameter determines the location of the resend request batching threshold in the receive window (Fig. 4). The parameter value is in per cent of a span that starts with the resend request threshold and ends with the front end of the receive window. The maximum allowed parameter value is 100. In Fig. 4 the resend request batching threshold is set at 80% of the span.

The resend request batching threshold relates to consecutive missing packets, i.e., to a continuous sequence of missing packets. Resend request batching allows grouping of consecutive missing packets for a single resend request rather than sending a sequence of resend requests where each resend request relates to just one missing packet.

The location of the resend request batching threshold determines the maximum number of consecutive missing packets that can be grouped together for a single resend request. The maximum number corresponds to the number of packets that fit into the span between the resend request threshold and the resend request batching threshold plus one.

If the Resend Request Batching parameter is set to 0, no batching will occur and a resend request will be sent for each single missing packet. For other settings, consider an example: Suppose the Resend Request Batching parameter is set to 80 referring to a span between the resend request threshold and the front end of the receive window that can hold five packets (Fig. 4). In this case 4 packets (5 x 80%) will fit into the span between the resend request threshold and the resend request batching threshold. Accordingly, the maximum number of consecutive missing packets that can be batched is 5 (4 + 1).

## Timeout Resend Mechanism Parameters

The timeout resend mechanism is illustrated in Fig. 5 where the following assumptions are made:

- The frame includes 3000 packets.
- Packet 1002 is missing within the stream of packets and has not been recovered.
- Packets 2999 and 3000 are missing at the end of the stream of packets (end of the frame).
- The Maximum Number Resend Requests parameter is set to 3.

DIAGRAM IS NOT DRAWN TO SCALE

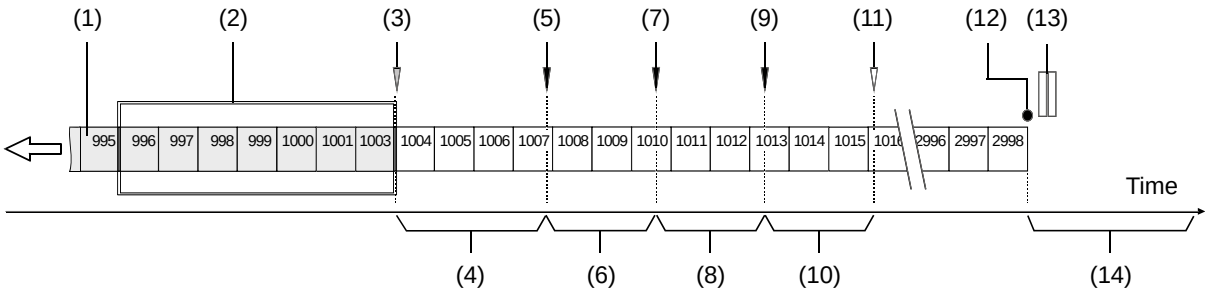



Fig. 5: Incomplete Stream of Packets and Part of the Resend Mechanism

(1) Stream of packets. Gray indicates that the status was checked as the packet entered the receive window. White indicates that the status has not yet been checked.

(2) Receive window of the performance driver.

(3) As packet 1003 enters the receive window, packet 1002 is detected as missing.

(4) Interval defined by the Resend Timeout parameter.

(5) The Resend Timeout interval expires and the first resend request for packet 1002 is sent to the camera. The camera does not respond with a resend.

(6) Interval defined by the Resend Response Timeout parameter.

(7) The Resend Response Timeout interval expires and a second resend request for packet 1002 is sent to the camera. The camera does not respond with a resend.

(8) Interval defined by the Resend Response Timeout parameter.

(9) The Resend Response Timeout interval expires and a third resend request for packet 1002 is sent to the camera. The camera still does not respond with a resend.

(10) Interval defined by the Resend Response Timeout parameter.

(11) Because the maximum number of resend requests has been sent and the last Resend Response Timeout interval has expired, packet 1002 is now considered as lost.

(12) End of the frame.

(13) Missing packets at the end of the frame (2999 and 3000).

(14) Interval defined by the Packet Timeout parameter.

**Maximum Number Resend Requests** - The Maximum Number Resend Requests parameter sets the maximum number of resend requests the performance driver will send to the camera for each missing packet.

**Resend Timeout** - The Resend Timeout parameter defines how long (in milliseconds) the performance driver will wait after detecting that a packet is missing before sending a resend request to the camera. The parameter applies only once to each missing packet after the packet was detected as missing.

**Resend Request Response Timeout** - The Resend Request Response Timeout parameter defines how long (in milliseconds) the performance driver will wait after sending a resend request to the camera before considering the resend request as lost.

If a resend request for a missing packet is considered lost and if the maximum number of resend requests as set by the Maximum Number Resend Requests parameter has not yet been reached, another resend request will be sent. In this case, the parameter defines the time separation between consecutive resend requests for a missing packet.

**Packet Timeout** - The Packet Timeout parameter defines how long (in milliseconds) the performance driver will wait for the next expected packet before it sends a resend request to the camera. This parameter ensures that resend requests are sent for missing packets near to the end of a frame. In the event of a major interruption in the stream of packets, the parameter will also ensure that resend requests are sent for missing packets that were detected to be missing immediately before the interruption. Make sure the Packet Timeout parameter is set to a longer time interval than the time interval set for the inter-packet delay.

## Threshold and Timeout Resend Mechanisms Combined

Fig. 6 illustrates the combined action of the threshold and the timeout resend mechanisms where the following assumptions are made:

- All parameters set to default.
- The frame includes 3000 packets.
- Packet 1002 is missing within the stream of packets and has not been recovered.
- Packets 2999 and 3000 are missing at the end of the stream of packets (end of the frame).

The default values for the performance driver parameters will cause the threshold resend mechanism to become operative before the timeout resend mechanism. This ensures maximum efficiency and that resend requests will be sent for all missing packets.

With the default parameter values, the resend request threshold is located very close to the front end of the receive window. Accordingly, there will be only a minimum delay between detecting a missing packet and sending a resend request for it. In this case, a delay according to the Resend Timeout parameter will not occur (see Fig. 6). In addition, resend request batching will not occur.

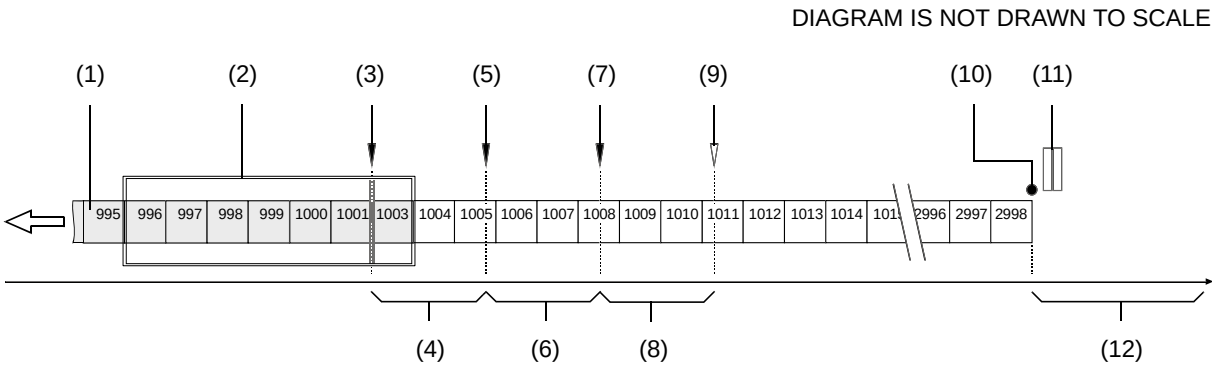



Fig. 6: Combination of Threshold Resend Mechanism and Timeout Resend Mechanism

(1) Stream of packets, Gray indicates that the status was checked as the packet entered the receive window. White indicates that the status has not yet been checked.

(2) Receive window of the performance driver.

(3) Threshold for sending resend requests (resend request threshold). The first resend request for packet 1002 is sent to the camera. The camera does not respond with a resend.

(4) Interval defined by the Resend Response Timeout parameter.

(5) The Resend Timeout interval expires and the second resend request for packet 1002 is sent to the camera. The camera does not respond with a resend.

(6) Interval defined by the Resend Response Timeout parameter

(7) The Resend Timeout interval expires and the third resend request for packet 1002 is sent to the camera. The camera does not respond with a resend.

(8) Interval defined by the Resend Response Timeout parameter

(9) Because the maximum number of resend requests has been sent and the last Resend Response Timeout interval has expired, packet 1002 is now considered as lost.

(10) End of the frame.

(11) Missing packets at the end of the frame (2999 and 3000).

(12) Interval defined by the Packet Timeout parameter.

You can set the performance driver parameter values from within your application software by using the Basler pylon API. The following code snippet illustrates using the API to read and write the parameter values:

```
// Get the Stream Parameters object
Camera_t::StreamGrabber_t StreamGrabber( Camera.GetStreamGrabber(0) );

// Write the ReceiveWindowSize parameter
StreamGrabber.ReceiveWindowSize.SetValue( 16 );

// Disable packet resends
StreamGrabber.EnableResend.SetValue( false );

// Write the PacketTimeout parameter
StreamGrabber.PacketTimeout.SetValue( 40 );

// Write the ResendRequestThreshold parameter
StreamGrabber.ResendRequestThreshold.SetValue( 5 );

// Write the ResendRequestBatching parameter
StreamGrabber.ResendRequestBatching.SetValue( 10 );

// Write the ResendTimeout parameter
StreamGrabber.ResendTimeout.SetValue( 2 );

// Write the ResendRequestResponseTimeout parameter
StreamGrabber.ResendRequestResponseTimeout.SetValue( 2 );

// Write the MaximumNumberResendRequests parameter
StreamGrabber.MaximumNumberResendRequests.SetValue( 25 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters. (Note that the performance driver parameters will only appear in the viewer if the performance driver is installed on the adapter to which your camera is connected.)

For more information about the pylon Viewer, see Section 3.1 on page 17.

## Adapter Properties

When the Basler Performance driver is installed, it adds a set of "advanced" properties to the network adapter. These properties include:

**Max Packet Latency** - A value in microseconds that defines how long the adapter will wait after it receives a packet before it generates a packet received interrupt.

**Max Receive Inter-packet Delay** - A value in microseconds that defines the maximum amount of time allowed between incoming packets.

**Maximum Interrupts per Second** - Sets the maximum number of interrupts per second that the adapter will generate.

**Network Address** - allows the user to specify a MAC address that will override the default address provided by the adapter.

**Packet Buffer Size** - Sets the size in bytes of the buffers used by the receive descriptors and the transmit descriptors.

**Receive Descriptors** - Sets the number of descriptors to use in the adapter's receiving ring.

**Transmit Descriptors** - Sets the number of descriptors to use in the adapter's transmit ring.

To access the advanced properties for an adapter:

1. Open a **Network Connections** window and find the connection for your network adapter.
2. Right click on the name of the connection and select **Properties** from the drop down menu.
3. A **LAN Connection Properties** window will open. Click the **Configure** button.
4. An **Adapter Properties** window will open. Click the **Advanced** tab.

**We strongly recommend using the default parameter settings.** Changing the parameters can have a significant negative effect on the performance of the adapter and the driver.

# 4.3 Transport Layer Parameters

The transport layer parameters are part of the camera's basic GigE implementation. These parameters do not normally require adjustment.

**Read Timeout** - If a register read request is sent to the camera via the transport layer, this parameter designates the time out (in milliseconds) within which a response must be received.

**Write Timeout** - If a register write request is sent to the camera via the transport layer, this parameter designates the time out (in milliseconds) within which an acknowledge must be received.

**Heartbeat Timeout** - The GigE Vision standard requires implementation of a heartbeat routine to monitor the connection between the camera and the host PC. This parameter sets the heartbeat timeout (in milliseconds). If a timeout occurs, the camera releases the network connection and enters a state that allows reconnection.

Management of the heartbeat time is normally handled by the Basler's basic GigE implementation and changing this parameter is not required for normal camera operation. However, if you are debugging an application and you stop at a break point, you will have a problem with the heartbeat timer. The timer will time out when you stop at a break point and the connection to the camera will be lost. When debugging, you should increase the heartbeat timeout to a high value to avoid heartbeat timeouts at break points. When debugging is complete, you should return the timeout to its normal setting.

You can set the driver related transport layer parameter values from within your application software by using the Basler pylon API. The following code snippet illustrates using the API to read and write the parameter values:

```
 // Read/Write Timeout
Camera_t::TlParams_t TlParams( Camera.GetTLNodeMap() );
TlParams.ReadTimeout.SetValue(500);  // 500 milliseconds
TlParams.WriteTimeout.SetValue(500); // 500 milliseconds

// Heartbeat Timeout
Camera_t::TlParams_t TlParams( Camera.GetTLNodeMap() );
TlParams.HeartbeatTimeout.SetValue(5000);  // 5 seconds
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

# 5 Network Related Camera Parameters and Managing Bandwidth

This section describes the camera parameters that are related to the camera's performance on the network. It also describes how to use the parameters to manage the available network bandwidth when you are using multiple cameras.

## 5.1 Network Related Parameters in the Camera

The camera includes several parameters that determine how it will use its network connection to transmit data to the host PC. The list below describes each parameter and provides basic information about how the parameter is used. The following section describes how you can use the parameters to manage the bandwidth used by each camera on your network.

**Payload Size** (read only)

Indicates the total size in bytes of the image data plus any chunk data (if chunks are enabled) in each frame that the camera will transmit. Packet headers are not included.

**Stream Channel Selector** (read/write)

The GigE Vision standard specifies a mechanism for establishing several separate stream channels between the camera and the PC. This parameter selects the stream channel that will be affected when the other network related parameters are changed.

Currently, the cameras support only one stream channel, i.e., stream channel 0.

**Packet Size** (read/write)

As specified in the GigE Vision standard, each acquired frame will be fit into a data block. The block contains three elements: a *data leader* consisting of one packet used to signal the beginning of a data block, the *data payload* consisting of one or more packets containing the actual data for the current block, and a *data trailer* consisting of one packet used to signal the end of the data block.

The packet size parameter sets the size of the packets that the camera will use when it sends the data payload via the selected stream channel. The value is in bytes. The value does not affect the leader and trailer size using a total of 36 bytes, and the last data packet may be a smaller size. The payload size will be the packet size minus 36 bytes.

The packet size parameter should always be set to the maximum size that your network adapter and network switches (if used) can handle.

**Inter-packet Delay** (read/write)

Sets the delay in ticks between the packets sent by the camera. Applies to the selected stream channel. Increasing the inter-packet delay will decrease the camera's effective data transmission rate and will thus decrease the network bandwidth used by the camera.

In the current camera implementation, one tick = 8 ns. To check the tick frequency, you can read the Gev Timestamp Tick Frequency parameter value. This value indicates the number of clock ticks per second.

When setting the time interval for the inter-packet delay, make sure that the time interval for the packet timeout is set to a higher value.

**Frame Transmission Delay** (read/write)

Sets a delay in ticks (one tick = 8 ns) between when a camera would normally begin transmitting an acquired frame and when it actually begins transmission. This parameter should be set to zero in most normal situations.

If you have many cameras in your network and you will be simultaneously triggering image acquisition on all of them, you may find that your network switch or network adapter is overwhelmed if all of the cameras simultaneously begin to transmit frame data at once. The frame transmission delay parameter can be used to stagger the start of frame data transmission from each camera.

**Bandwidth Assigned** (read only)

Indicates the bandwidth in bytes per second that will be used by the camera to transmit frame and chunk feature data and to handle resends and control data transmissions. The value of this parameter is a result of the packet size and the inter-packet delay parameter settings.

In essence, the bandwidth assigned is calculated this way:

$$\text{Bandwidth Assigned} = \frac{\frac{\text{X Packets}}{\text{Frame}} \times \frac{\text{Y Bytes}}{\text{Packet}}}{\left[\frac{\text{X Packets}}{\text{Frame}} \times \frac{\text{Y Bytes}}{\text{Packet}} \times \frac{8\ \text{ns}}{\text{Byte}}\right] + \left[\left(\frac{\text{X Packets}}{\text{Frame}} - 1\right) \times (\text{IPD} \times 8\ \text{ns})\right]}$$

Where: X = number of packets needed to transmit the frame

Y = number of bytes in each packet

IPD = Inter-packet Delay setting in ticks (with a tick set to the 8 ns standard)

When considering this formula, you should know that on a Gigabit network it takes one tick to transmit one byte. Also, be aware that the formula has been simplified for easier understanding.

For more information about the pylon Viewer, see Section 3.3 on page 18.

# 5.2 Managing Bandwidth When Multiple Cameras Share a Single Network Path

If you are using a single camera on a GigE network, the problem of managing bandwidth is simple. The network can easily handle the bandwidth needs of a single camera and no intervention is required. A more complicated situation arises if you have multiple cameras connected to a single network adapter as shown in Fig. 7.

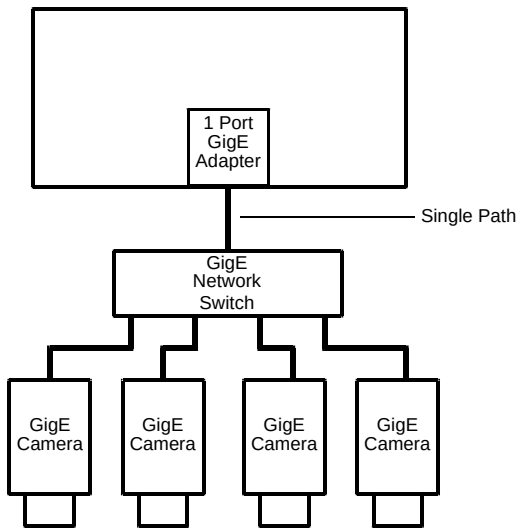



Fig. 7: Multiple Cameras on a Network

One way to manage the situation where multiple cameras are sharing a single network path is to make sure that only one of the cameras is acquiring and transmitting frames at any given time. The data output from a single camera is well within the bandwidth capacity of the single path and you should have no problem with bandwidth in this case.

If you want to acquire and transmit frames from several cameras simultaneously, however, you must determine the total data output rate for all the cameras that will be operating simultaneously and you must make sure that this total does not exceed the bandwidth of the single path (125 MByte/s).

An easy way to make a quick check of the total data output from the cameras that will operate simultaneously is to read the value of the Bandwidth Assigned parameter for each camera. This parameter indicates the camera’s gross data output rate in bytes per second with its current settings. If the sum of the bandwidth assigned values is less than 125 MByte/s, the cameras should be able to operate simultaneously without problems. If it is greater, you must lower the data output rate of one or more of the cameras.

You can lower the data output rate on a camera by using the Inter-packet Delay parameter. This parameter adds a delay between the transmission of each packet from the camera and thus slows the data transmission rate of the camera. The higher the inter-packet delay parameter is set, the greater the delay between the transmission of each packet will be and the lower the data

transmission rate will be. After you have adjusted the Inter-packet Delay parameter on each camera, you can check the sum of the Bandwidth Assigned parameter values and see if the sum is now less than 125 MByte/s.

## 5.2.1 A Procedure for Managing Bandwidth

In theory, managing bandwidth sharing among several cameras is as easy as adjusting the inter-packet delay. In practice, it is a bit more complicated because you must consider several factors when managing bandwidth. The procedure below outlines a structured approach to managing bandwidth for several cameras.

The objectives of the procedure are:

- To optimize network performance.
- To determine the bandwidth needed by each camera for frame data transmission.
- To determine the bandwidth actually assigned to each camera for frame data transmission.
- For each camera, to make sure that the actual bandwidth assigned for frame data transmission matches the bandwidth needed.
- To make sure that the total bandwidth assigned to all cameras does not exceed the network's bandwidth capacity.
- To make adjustments if the bandwidth capacity is exceeded.

**Step 1 - Optimize the Network Performance.**

If, as recommended, you are using the Basler performance driver with an Intel PRO network adapter or a compatible network adapter, the network parameters for the network adapter are automatically optimized and need not be changed. Go on to step two now.

If you are using the Basler filter driver and you have already set the network parameters for your adapter during the installation of the Basler pylon software, go on to step two now.

Otherwise, open the **Network Connection Properties** window for your network adapter and check the following network parameters:

- If you are using an Intel PRO network adapter, make sure the **Receive Descriptors** parameter is set to its maximum value and the **Interrupt Moderation Rate** parameter is set to **Extreme**.

  Also make sure the **Speed and Duplex Mode** parameter is set to **Auto Detect**.
- If you are using a different network adapter, see whether parameters are available that will allow you to set the number of receive descriptors and the number of CPU interrupts. The related parameter names may differ from the ones used for the Intel PRO adapters. Also, the way of setting the parameters may be different. You may, for example, need to use a parameter to set a low number for the interrupt moderation and then use a different parameter to enable the interrupt moderation.

  If possible, set the number of receive descriptors to a maximum value and set the number of CPU interrupts to a low value.

  If possible, also set the parameter for speed and duplex to auto detect.

Contact Basler technical support if you need further assistance.

**Step 2 - Set the Packet Size parameter on each camera as large as possible.**

Using the largest possible packet size has two advantages, it increases the efficiency of network transmissions between the camera and the PC and it reduces the time required by the PC to process incoming packets. The largest packet size setting that you can use with your camera is determined by the largest packet size that can be handled by your network. The size of the packets that can be handled by the network depends on the capabilities and settings of the network adapter you are using and on capabilities of the network switch you are using.

Start by checking the documentation for your adapter to determine the maximum packet size (sometimes called “frame” size) that the adapter can handle. Many adapters can handle what is known as “jumbo packets” or "jumbo frames". These are packets with a 16 kB size. Once you have determined the maximum size packets the adapter can handle, make sure that the adapter is set to use the maximum packet size.

Next, check the documentation for your network switch and determine the maximum packet size that it can handle. If there are any settings available for the switch, make sure that the switch is set for the largest packet size possible.

Now that you have set the adapter and switch, you can determine the largest packet size the network can handle. The device with the smallest maximum packet size determines the maximum allowed packet size for the network. For example, if the adapter can handle 16 kB packets and the switch can handle 8 kB packets, then the maximum for the network is 8 kB packets.

Once you have determined the maximum packet size for your network, set the value of the Packet Size parameter on each camera to this value.

**Tip**

The manufacturer’s documentation sometimes makes it difficult to determine the maximum packet size for a device, especially network switches. There is a "quick and dirty" way to check the maximum packet size for your network with its current configuration:

1. Open the pylon Viewer, select a camera, and set the Packet Size parameter to a low value (1 kB for example).
2. Use the Continuous Shot mode to acquire several frames.
3. Gradually increase the value of the Packet Size parameter and acquire a few frames after each size change.
4. When your Packet Size setting exceeds the packet size that the network can handle, the pylon Viewer will lose the ability to acquire frames. (When you use Continuous Shot, the Viewer’s status bar will indicate that it is acquiring frames, but the frame in the viewing area will appear to be frozen.)

**Step 3 - Set the Bandwidth Reserve parameter for each camera.**

The Bandwidth Reserve parameter setting for a camera determines how much of the bandwidth assigned to that camera will be reserved for lost packet resends and for asynchronous traffic such as commands sent to the camera. If you are operating the camera in a relatively EMI free environment, you may find that a bandwidth reserve of 2% or 3% is adequate. If you are operating in an extremely noisy environment, you may find that a reserve of 8% or 10% is more appropriate.

**Step 4 - Calculate the "data bandwidth needed" by each camera.**

The objective of this step is to determine how much bandwidth (in Byte/s) each camera needs to transmit the frame data that it generates. The amount of data bandwidth a camera needs is the product of several factors: the amount of image data included in each frame, the amount of chunk data being added to each frame, the "packet overhead" such as packet leaders and trailers, and the number of frames the camera is acquiring each second.

For each camera, you can use the two formulas below to calculate the data bandwidth needed. To use the formulas, you will need to know the current value of the Payload Size parameter and the Packet Size parameter for each camera. You will also need to know the frame rate (in frames/s) at which each camera will operate.

$$\text{Bytes/Frame} = \left\lfloor \left\lceil \frac{\text{Payload Size}}{\text{Packet Size}} \right\rceil_1 \times \text{Packet Overhead} \right\rfloor + \lceil \text{Payload Size} \rceil_4 + \text{Leader Size} + \text{Trailer Size}$$

$$\text{Data Bandwidth Needed} = \text{Bytes/Frame} \times \text{Frames/s}$$

Where:

Packet Overhead = 72 (for a GigE network)
78 (for a 100 MBit/s network)

Leader Size = Packet Overhead + 36 (if chunk mode is not active)
Packet Overhead + 12 (if chunk mode is active)

Trailer Size = Packet Overhead + 8

$\lceil x \rceil_1$ means round up x to the nearest integer

$\lceil x \rceil_4$ means round up x to the nearest multiple of 4

**Step 5 - Calculate “data bandwidth assigned” to each camera.**

For each camera, there is a parameter called Bandwidth Assigned. This read only parameter indicates the total bandwidth that has been assigned to the camera. The Bandwidth Assigned parameter includes both the bandwidth that can be used for frame data transmission plus the bandwidth that is reserved for packet resents and camera control signals. To determine the “data bandwidth assigned,” you must subtract out the reserve.

You can use the formula below to determine the actual amount of assigned bandwidth that is available for data transmission. To use the formula, you will need to know the current value of the Bandwidth Assigned parameter and the Bandwidth reserve parameter for each camera.

$$\text{Data Bandwidth Assigned} = \text{Bandwidth Assigned} \times \frac{100 - \text{Bandwidth Reserved}}{100}$$

**Step 6 - For each camera, compare the data bandwidth needed with the data bandwidth assigned.**

For each camera, you should now compare the data bandwidth assigned to the camera (as determined in step 4) with the bandwidth needed by the camera (as determined in step 3).

For bandwidth to be used most efficiently, the data bandwidth assigned to a camera should be equal to or just slightly greater than the data bandwidth needed by the camera. If you find that this is the situation for all of the cameras on the network, you can go on to step 6 now. If you find a camera that has much more data bandwidth assigned than it needs, you should make an adjustment.

To lower the amount of data bandwidth assigned, you must adjust a parameter called the Inter-packet Delay. If you increase the Inter-packet Delay parameter value on a camera, the data bandwidth assigned to the camera will decrease. So for any camera where you find that the data bandwidth assigned is much greater then the data bandwidth needed, you should do this:

1. Raise the setting for the Inter-packet delay parameter for the camera.
2. Recalculate the data bandwidth assigned to the camera.
3. Compare the new data bandwidth assigned to the data bandwidth needed.
4. Repeat 1, 2, and 3 until the data bandwidth assigned is equal to or just greater than the data bandwidth needed.

**Note**

If you increase the inter-packet delay to lower a camera’s data output rate there is something that you must keep in mind. When you lower the data output rate, you increase the amount of time that the camera needs to transmit an acquired frame. Increasing the frame transmission time can restrict the camera’s maximum allowed acquisition line rate.

**Step 7 - Check that the total bandwidth assigned is less than the network capacity.**

1. For each camera, determine the current value of the Bandwidth Assigned parameter. The value is in Byte/s. (Make sure that you determine the value of the Bandwidth Assigned parameter after you have made any adjustments described in the earlier steps.)
2. Find the sum of the current Bandwidth Assigned parameter values for all of the cameras.

If the sum of the Bandwidth Assigned values is less than 125 MByte/s for a GigE network or 12.5 M/Byte/s for a 100 Bit/s network, the bandwidth management is OK.

If the sum of the Bandwidth Assigned values is greater than 125 MByte/s for a GigE network or 12.5 M/Byte/s for a 100 Bit/s network, the cameras need more bandwidth than is available and you must make adjustments. In essence, you must lower the data bandwidth needed by one or more of the cameras and then adjust the data bandwidths assigned so that they reflect the lower bandwidth needs.

You can lower the data bandwidth needed by a camera either by lowering its line rate or by decreasing the size of the frame. Once you have adjusted the line rates and/or frame size on the cameras, you should repeat steps 2 through 6.

For more information about the camera’s maximum allowed line rate, see Section 8.7 on page 129.

For more information about the frame size, see Section 8.1 on page 71.

# 6 Camera Functional Description

This chapter provides an overview of the camera's functionality from a system perspective. The overview will aid your understanding when you read the more detailed information included in the later chapters of the user's manual.

Each camera employs a single line CMOS sensor chip designed for monochrome imaging. For 2k cameras, the sensor includes 2048 pixels. For 4k cameras, the sensor consists of two 2k sensor segments, resulting in a total of 4096 pixels.

Acquisition start, frame start, and line start can be controlled via externally generated hardware trigger signals. These signals facilitate periodic or non-periodic frame/line start. Modes are available that allow the length of exposure time to be directly controlled by the external line start signal or to be set for a pre-programmed period of time.

Acquisition start, frame start, and exposure time can also be controlled by parameters transmitted to the camera via the Basler pylon API and the GigE interface.

Accumulated charges are read out of the sensor when exposure ends. At readout, accumulated charges are moved from the sensor's light-sensitive elements (pixels) into the analog processing section of the sensor (Fig. 8 on page 38). As the charges move from the pixels to the analog processing section, they are converted to voltages proportional to the size of each charge. The voltages from the analog processing section are next passed to a bank of 12 Bit Analog-to-Digital converters (ADCs).

Finally, the gray values pass through a section of the sensor where they receive additional digital processing and then they are moved out of the sensor. As each gray values leaves the sensor, it passes through an FPGA and into an image buffer (Fig. 9 on page 38). All shifting is clocked according to the camera's internal data rate. Shifting continues until all image data has been read out of the sensor.

The gray values leave the image buffer and passes back through the FPGA to an Ethernet controller where it is assembled into data packets. The packets are then transmitted via an Ethernet network to a network adapter in the host PC. The Ethernet controller also handles transmission and receipt of control data such as changes to the camera's parameters.

The image buffer between the sensor and the Ethernet controller allows data to be read out of the sensor at a rate that is independent of the data transmission rate between the camera and the host computer. This ensures that the data transmission rate has no influence on image quality.

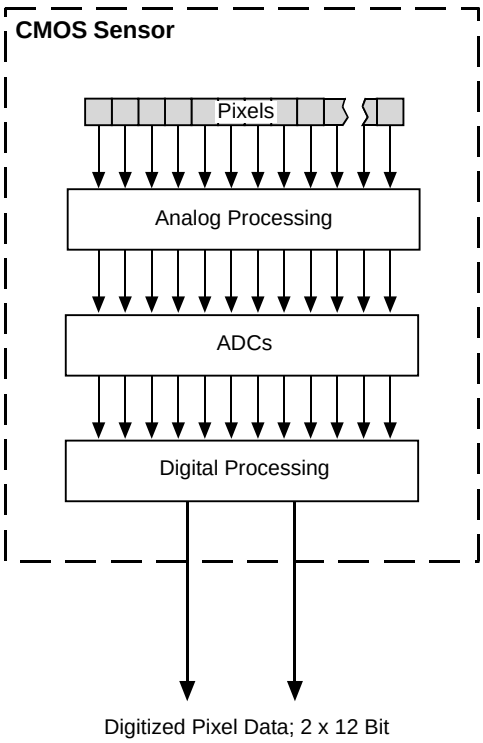



Fig. 8: CMOS Sensor Architecture for a 2k Sensor or 2k Sensor Segment with 7 µm x 7 µm Pixel Size

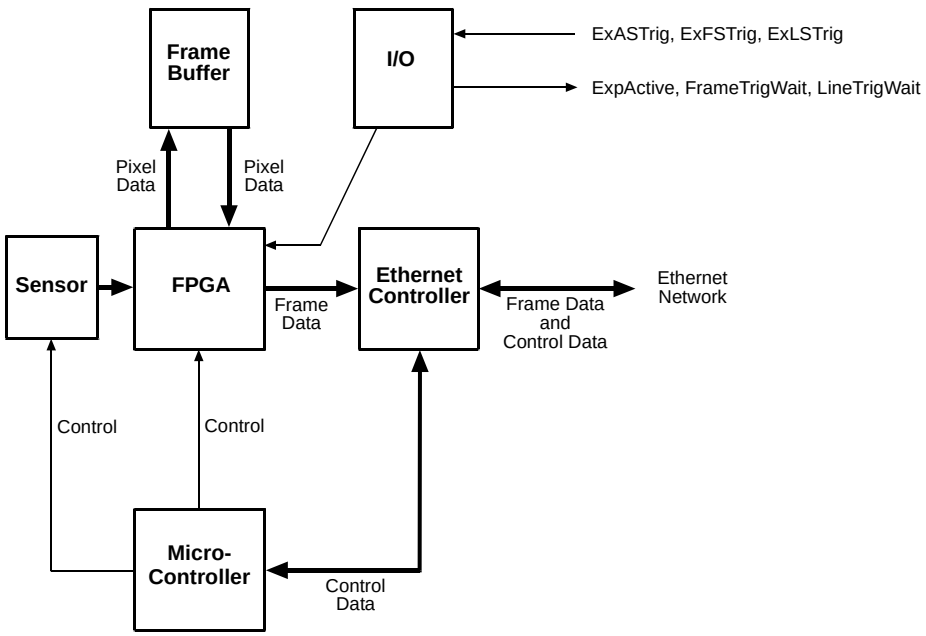



Fig. 9: Camera Block Diagram

# 7 Physical Interface

This chapter provides detailed information, such as pinouts and voltage requirements, for the physical interface on the camera. This information will be especially useful during your initial design-in process.

## 7.1 General Description of the Connections

The camera is interfaced to external circuity via connectors located on the back of the housing:

- A 6-pin receptacle used to provide power to the camera.
- A 12-pin receptacle used to provide access to the camera's I/O lines.
- An 8-pin RJ-45 jack used to provide a 100/1000 Mbit/s Ethernet connection to the camera. The jack includes a green LED and a yellow LED that indicate the state of the network connection.

The drawing below shows the location of the three connectors and the LEDs.

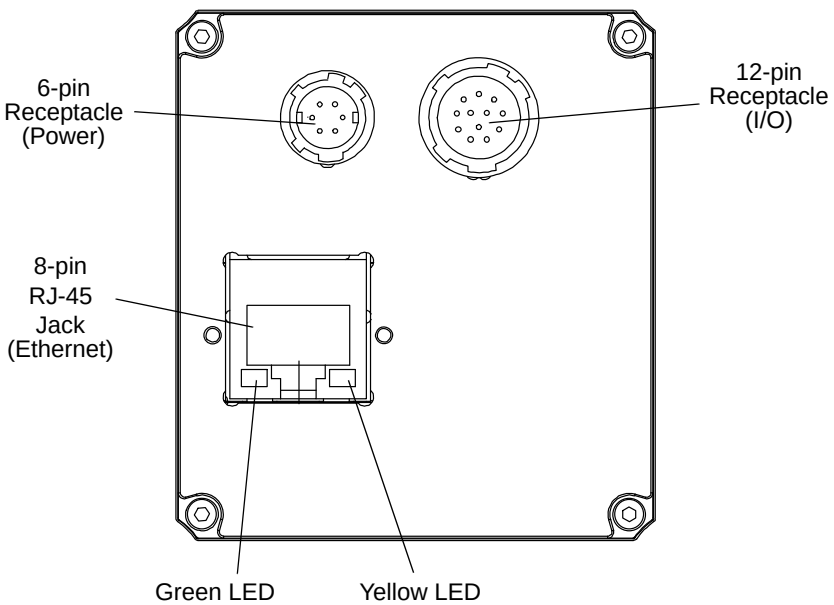



Fig. 10: Camera Connectors and LEDs

### 7.1.1 Pin Numbering

Pin numbering for the camera's 6-pin and 12-pin receptacles is as shown in Fig. 11.

Pin numbering for the 8-pin RJ-45 jack adheres to the Ethernet standard.

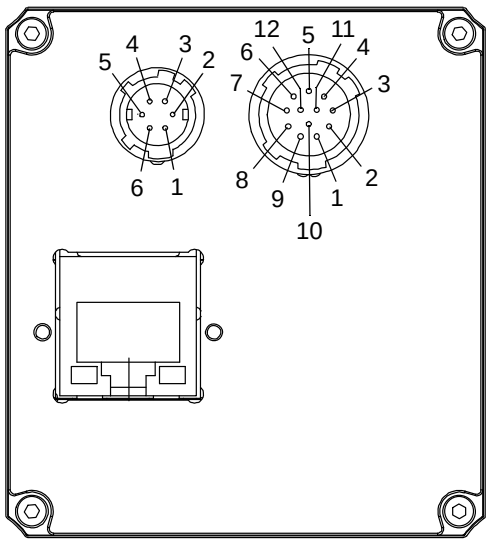



Fig. 11: Pin Numbering for the 6-pin and 12-pin Receptacles

# 7.2 Connector Pin Assignments

## 7.2.1 Pin Assignments for the 6-pin Receptacle

The 6 pin receptacle is used to supply power to the camera. The pin assignments for the receptacle are shown in Table 2.

| Pin | Designation |
|---|---|
| 1 | +12 VDC Camera Power (+12 VDC ± 10%) * |
| 2 | +12 VDC Camera Power (+12 VDC ± 10%) * |
| 3 | Not Connected |
| 4 | Not Connected |
| 5 | DC Ground ** |
| 6 | DC Ground ** |

Table 2: Pin Assignments for the 6-pin Receptacle

***NOTICE***

- Applying incorrect power can damage the camera.

  The camera‘s required nominal operating voltage is +12 VDC (±10 %), effective on the camera‘s connector.
- Applying power with the wrong polarity can severely damage the camera.

  Make sure that the polarity of the power applied to the camera is correct. Applying power with the wrong polarity can severely damage the camera.

* Pins 1 and 2 are tied together inside of the camera.

** Pins 5 and 6 are tied together inside of the camera.

To avoid a voltage drop when there are long wires between your power supply and the camera, we recommend that you provide +12 VDC through two separate wires between the power supply and pins 1 and 2 in the receptacle. We also recommend that you provide the ground through two separate wires between the power supply and pins 5 and 6.

## 7.2.2 Pin Assignments for the 12-pin Receptacle

The 12 pin receptacle is used to access the three physical input lines and two physical output lines on the camera. The pin assignments for the receptacle are shown in Table 3.

| Pin | Designation |
|---|---|
| 1 | I/O Input 1 - |
| 2 | I/O Input 1+ |
| 3 | I/O Input 3 - |
| 4 | I/O Input 3 + |
| 5 | Gnd |
| 6 | I/O Output 1- |
| 7 | I/O Output 1 + |
| 8 | I/O Input 2 - |
| 9 | I/O Input 2 + |
| 10 | Not connected |
| 11 | I/O Output 2 - |
| 12 | I/O Output 2+ |

Table 3: Pin Assignments for the 12-pin Receptacle

***NOTICE***

- Using a wrong pin assignment for the 12-pin receptacle can severely damage the camera.

  Make sure the cable and plug you connect to the 12-pin receptacle follows the correct pin assignment. In particular, do **not** use a pin assignment that would be correct for Basler area scan cameras. The 12-pin receptacle of Basler line scan and area scan cameras are electrically incompatible.

For the I/O lines to work correctly, pin 5 must be connected to ground.

### 7.2.3 Pin Assignments for the RJ-45 Jack

The 8-pin RJ-45 jack provides Ethernet access to the camera. Pin assignments adhere to the Ethernet standard.

# 7.3 Connector Types

## 7.3.1 6-pin Connector

The 6-pin connector on the camera is a Hirose micro receptacle (part number HR10A-7R-6PB) or the equivalent.

The recommended mating connector is the Hirose micro plug (part number HR10A-7P-6S) or the equivalent.

## 7.3.2 12-pin Connector

The 12-pin connector on the camera is a Hirose micro receptacle (part number HR10A-10R-12P) or the equivalent.

The recommended mating connector is the Hirose micro plug (part number HR10A-10P-12S) or the equivalent.

## 7.3.3 RJ-45 Jack

The 8-pin jack for the camera's Ethernet connection is a standard RJ-45 connector.
The recommended mating connector is any standard RJ-45 plug.

Cables terminated with screw-lock connectors are available from Basler. Contact your Basler sales representative to order cable assemblies. Suitable cable assemblies are also available from the Intercon 1 division of Nortech Systems, Inc.

To ensure that you order cables with the correct connectors, note the horizontal orientation of the screws before ordering.

**Green and Yellow LEDs**

This RJ-45 jack on the camera includes a green LED and a yellow LED. When the green LED is lit, it indicates that an active network connection is available. When the yellow LED is lit, it indicates that data is being transmitted via the network connection.

# 7.4 Cabling Requirements

## 7.4.1 Power Cable

A single power cable is used to supply power to the camera.

The end of the power cable that connects to the camera's 6-pin connector must be terminated with a Hirose micro plug (part number HR10A-7P-6S) or the equivalent. The cable must be wired as shown in Fig. 12.

For proper EMI protection, the power cable terminated with the Hirose connector and attached to the camera must be a twin-cored, shielded cable. Also, the Hirose plug must be connected to the cable shield and the shield must be connected to earth ground at the power supply.

Close proximity to strong electromagnetic fields should be avoided.

***NOTICE***

An incorrect plug can damage the 6-pin connector.

The plug on the cable that you attach to the camera's 6-pin connector must have 6 female pins.

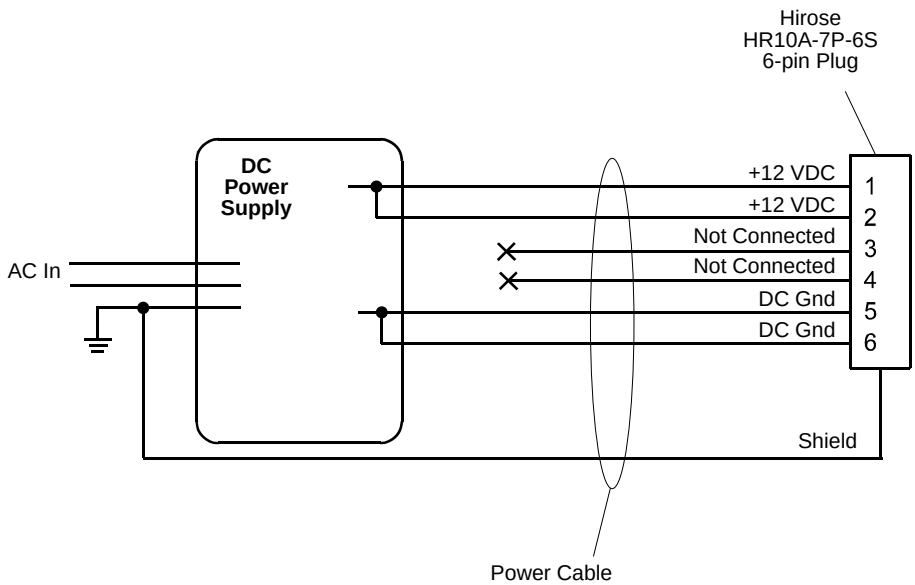



Fig. 12: Power Cable

## 7.4.2 I/O Cable

The end of the I/O cable that connects to the camera's 12-pin connector must be terminated with a Hirose micro plug (part number HR10A-10P-12S) or the equivalent. The cable must be wired as shown in Fig. 13.

The maximum length of the I/O cable is 10 meters, however, we strongly recommend keeping I/O cables as short as possible. The cable must be shielded and must be constructed with twisted pair wire. Use of twisted pair wire is essential to ensure that input signals are correctly received.

The required 12-pin Hirose plug is available from Basler. Basler also offers an I/O cable assembly that is terminated with a 12-pin Hirose plug on one end and unterminated on the other. Contact your Basler sales representative to order connectors or I/O cables.

Close proximity to strong electromagnetic fields should be avoided.

***NOTICE***

An incorrect plug can damage the 12-pin connector.

The plug on the cable that you attach to the camera's 12-pin connector must have 12 female pins.

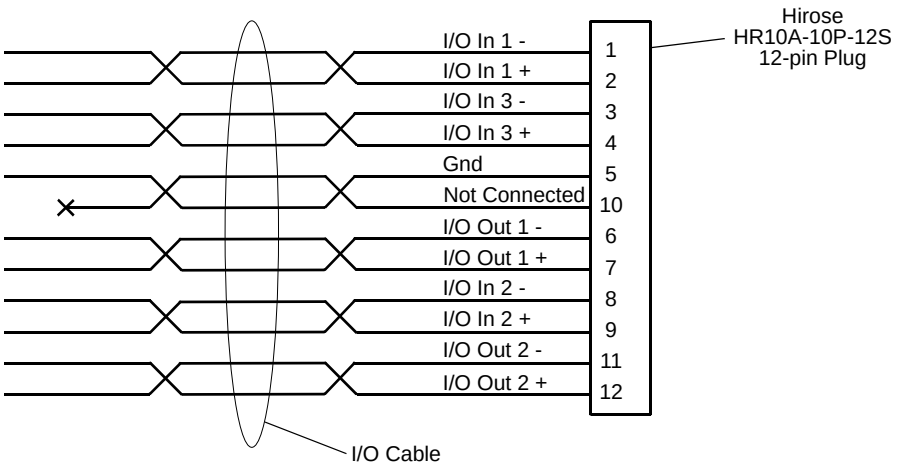



Fig. 13: I/O Cable

## 7.4.3 Ethernet Cables

Use high-quality Ethernet cables. To avoid EMI, the cables must be shielded. Use of category 6 or category 7 cables with S/STP shielding is strongly recommended. As a general rule, applications with longer cables or applications in harsh EMI conditions require higher category cables.

Either a straight-through (patch) or a cross-over Ethernet cable can be used to connect the camera directly to a GigE network adapter in a PC or to a GigE network switch.

Close proximity to strong electromagnetic fields should be avoided.

# 7.5 Camera Power

Camera power must be supplied to the 6-pin connector on the camera via a cable from your power supply. Nominal operating voltage is +12 VDC (± 10%) with less than one percent ripple. Power consumption is as shown in the specification tables in Section 1 of this manual.

Close proximity to strong electromagnetic fields should be avoided.

**NOTICE**

- Applying incorrect power can damage the camera.

  The camera‘s required nominal operating voltage is +12 VDC (±10 %), effective on the camera‘s connector.
- Applying power with the wrong polarity can severely damage the camera.

  Make sure that the polarity of the power applied to the camera is correct. Applying power with the wrong polarity can severely damage the camera.

**NOTICE**

An incorrect plug can damage the 6-pin connector.

The plug on the cable that you attach to the camera’s 6-pin connector must have 6 female pins.

For more information about the 6-pin connector, see Section 7.2.1 on page 41 and Section 7.3.1 on page 44.

For more information about the power cable, see Section 7.4.1 on page 45.

# 7.6 Input and Output Lines

## 7.6.1 Input Lines

The camera is equipped with three physical input lines designated as Input Line 1, Input Line 2, and Input Line 3. The input lines are accessed via the 12-pin connector on the back of the camera. The inputs are designed to accept RS-422 differential signals, but they can also be used with RS-644 low voltage differential signals or low voltage TTL signals.

### 7.6.1.1 Electrical Characteristics

#### Using the Inputs with RS-422

As shown in Fig. 14 and in the I/O schematic at the beginning of this section, each input is designed to receive an RS-422 signal. For the camera's I/O circuitry to operate properly, you must supply a ground as shown in Fig. 14.

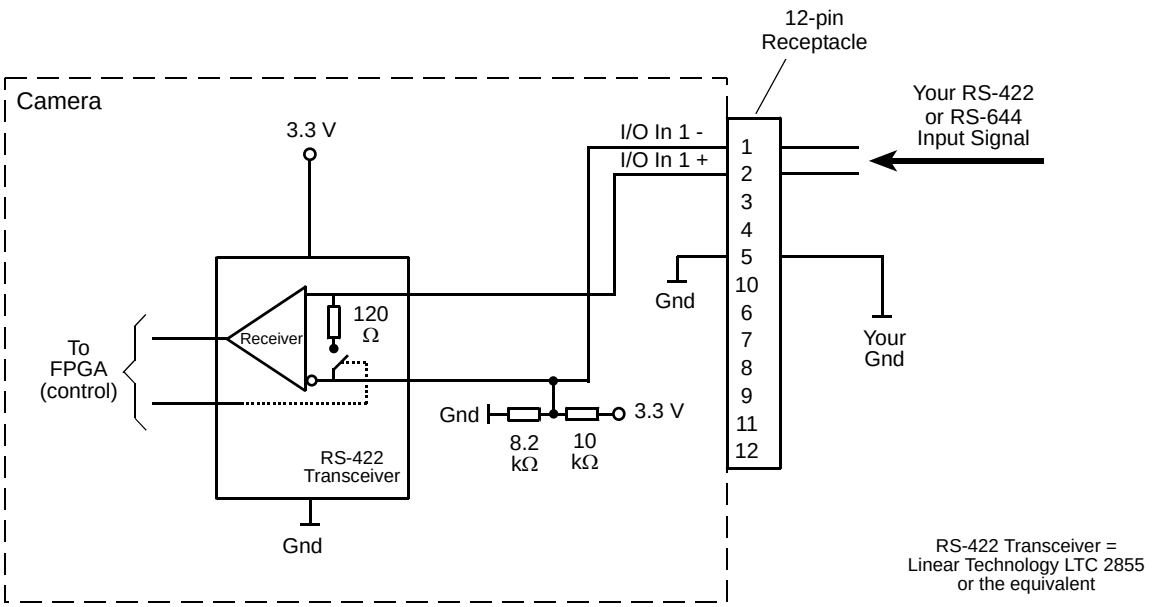



Fig. 14: Inputting RS-422 or RS-644 Signals

The RS-422 standard allows devices to be used with a bus structure to form an interface circuit. So, for example, input line 1 on several different cameras can be connected via an RS-422 bus as shown in Fig. 15.

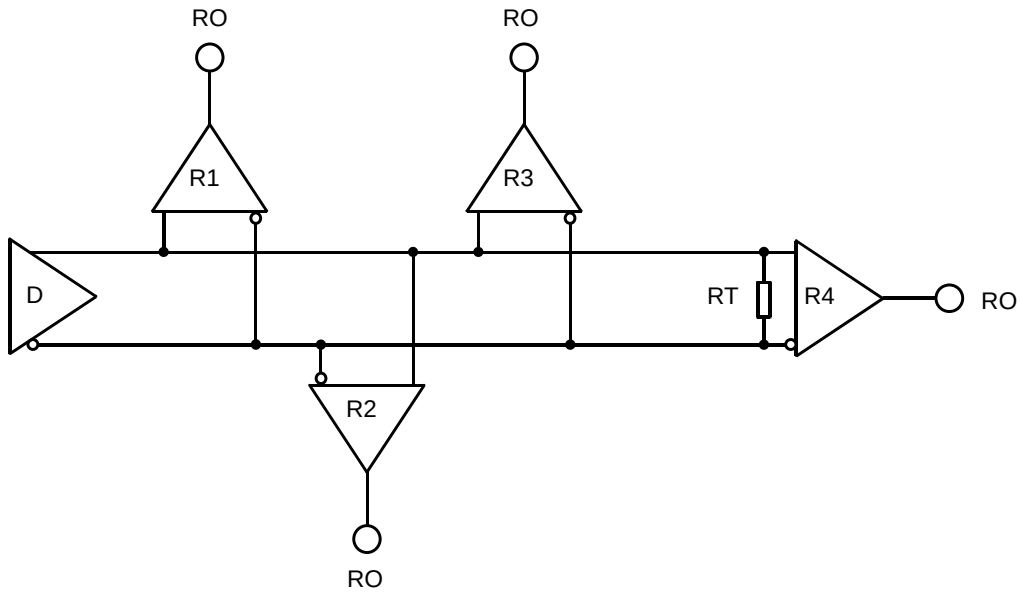



Fig. 15: RS-422 Interface Circuit Including Four Receivers as an Example

Connected to the bus would be one camera as the "master" transmitter (driver D; only one driver allowed) and up to ten cameras (receivers R), with the "master" transmitter sending signals to the "slave" inputs of the receivers. The inputs of the receivers would be connected in parallel to the driver via the bus.

The separations between receivers and bus should be as small as possible. The bus must be terminated by a 120 ohm termination resistor (RT). Note that each RS-422 input on the cameras includes a switchable 120 ohm termination resistor as shown in Fig. 14. When a camera input of the last receiver in the bus terminates the bus (as shown in Fig. 15.: R4), the termination resistor on that input should be enabled. You should not use multiple termination resistors on a single bus. Using multiple termination resistors will lower signalling reliability and has the potential for causing damage to the RS-422 devices.

## Using the Inputs with RS-644 LVDS

The inputs on the camera can accept RS-644 low voltage differential signals (LVDS).

If you are supplying an RS-644 LVDS signal to an input on the camera, the 120 ohm termination resistor on that input **must** be enabled. The input will not reliably react to RS-644 signals if the resistor is disabled.

For the camera's I/O circuitry to operate properly, you must supply a ground as shown in Fig. 14

| | |
|---|---|
| | Although the RS-644 standard allows several devices to be connected together in a "multidrop" configuration, we strongly recommend that you do not include any camera input in an RS-644 multidrop. Instead, we strongly recommend that you use a direct, point-to-point connection between your RS-644 transmitter and the camera input. |

## Using the Inputs with LVTTL

A camera input line can accept a Low Voltage TTL signal when the signal is input into the camera as shown in Fig. 16.

The following voltage requirements apply to the camera's I/O input (pin 2 of the 12-pin receptacle):

| Voltage | Significance |
|---|---|
| +0 to + 5.0 VDC | Recommended operating voltage. |
| +0 to +0.8 VDC | The voltage indicates a logical 0. |
| > +0.8 to +2.0 VDC | Region where the transition threshold occurs; the logical state is not defined in this region. |
| > +2.0 VDC | The voltage indicates a logical 1. |
| +6.0 VDC | Absolute maximum; the camera may be damaged when the absolute maximum is exceeded. |

Table 4: Voltage Requirements for the I/O Input When Using LVTTL

When LVTTL signals are applied to an input, the 120 ohm termination resistor on that input **must** be disabled. The input will not react to LVTTL signals if the resistor is enabled.

For the camera's I/O circuitry to operate properly, you must supply a ground as shown Fig. 16.

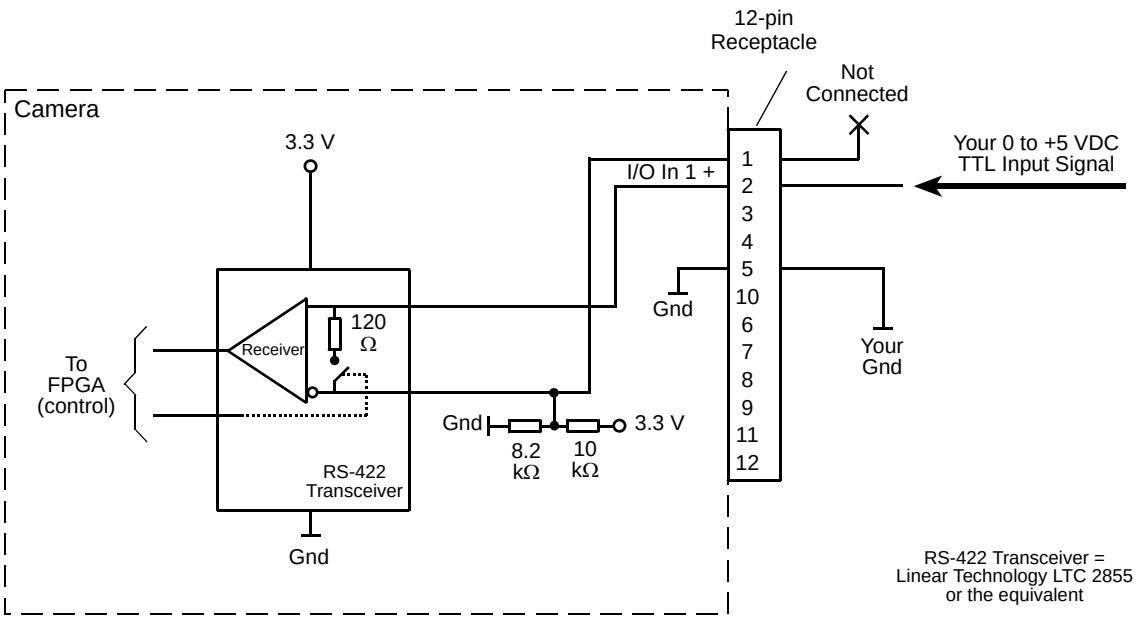



Fig. 16: Inputting Low Voltage TTL Signals

### Enabling and Disabling the Termination Resistor

You can select an input line and enable or disable the termination resistor on the line from within your application software by using the pylon API. The following code snippet illustrates using the API to set the parameter values:

```
Camera.LineSelector.SetValue( LineSelector_Line1 );
Camera.LineTermination.SetValue( true );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily enable or disable the resistors.

For more information about the pylon Viewer, see Section 3.1 on page 17.

## 7.6.1.2 Input Line Debouncers

Each individual input line is equipped with a debouncer. The debouncer aids in discriminating between valid and invalid input signals. The debouncer value specifies the minimum time that an input signal must remain high or remain low in order to be considered a valid input signal.

We recommend setting the debouncer value so that it is slightly greater than the longest expected duration of an invalid signal.

Setting the debouncer to a value that is too short will result in accepting invalid signals. Setting the debouncer to a value that is too long will result in rejecting valid signals.

The duration of a debouncer is determined by the value of the Line Debouncer Time Abs parameter value. The parameter is set in microseconds and can be set in a range from 0 to approximately 1 s.

To set a debouncer:

- Use the Line Selector to select the camera input line for which you want to set the debouncer.
- Set the value of the Line Debouncer Time Abs parameter.

You can set the Line Selector and the value of the Line Debouncer Abs parameter from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Select input line 1 and set the debouncer value to 100 microseconds
Camera.LineSelector.SetValue( LineSelector_Line1 );
Camera.LineDebouncerTimeAbs.SetValue( 100 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

### 7.6.1.3 Input Line Inverters

You can set each individual input line to invert or not to invert the incoming electrical signal. To set the invert function on an input line:

- Use the Line Selector to select an input line.
- Set the value of the Line Inverter parameter to true to enable inversion on the selected line and to false to disable inversion.

You can set the Line Selector and the Line Inverter parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Enable the inverter on line 1
Camera.LineSelector.SetValue( LineSelector_Line1 );
Camera.LineInverter.SetValue( true );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

### 7.6.1.4 Selecting an Input Line as a Source Signal for a Camera Function

You can select an input line as the source signal for the following camera functions:

- the Acquisition Start Trigger
- the Frame Start Trigger
- the Line Start Trigger
- the Phase A input for the shaft encoder module
- the Phase B input for the shaft encoder module

Note that to use an input line as the source signal for a camera function, you must apply an electrical signal to the input line that is appropriately timed for the function.

For detailed information about selecting an input line as the source signal for the camera's Acquisition Start Trigger function, see Section 8.2.2.2 on page 79.

For detailed information about selecting an input line as the source signal for the camera's Frame Start Trigger function, see Section 8.2.3.3 on page 85.

For detailed information about selecting an input line as the source signal for the camera's Line Start Trigger function, see Section 8.2.4.2 on page 88 and Section 8.2.4.3 on page 92.

For detailed information about selecting an input line as the source signal for the shaft encoder model Phase A or Phase B input, see Section 8.3 on page 114.

## Default Input Line Selections

By default:

- Input Line 1 is selected as the source signal for the camera's Line Start Trigger function.
- Input Line 1 is also selected as the source signal for shaft encoder module Phase A input.
- Input Line 2 is selected as the source signal for shaft encoder module Phase B input.
- Input Line 3 is selected as the source signal for the camera's Frame Start Trigger function.

# 7.6.2 Output Lines

The camera is equipped with two physical output lines designated as Output Line 1 and Output Line 2. The output lines are accessed via the 12-pin connector on the back of the camera. The outputs are designed to transmit RS-422 differential signals, but they can also be used with RS-644 low voltage differential signalling or low voltage TTL signalling.

## 7.6.2.1 Electrical Characteristics

### Using the Outputs with RS-422

As shown in Fig. 17 and in the I/O schematic at the beginning of this section, each output is designed to transmit an RS-422 signal. For the camera's I/O circuitry to operate properly, you must supply a ground as shown in Fig. 17.

The RS-422 standard allows devices to be used with a bus structure to form an interface circuit. So, for example, output line 1 on a camera can be connected to an RS-422 bus in parallel with the inputs on several of your devices (receivers). The camera with output line 1 connected to the bus would serve as a "master" transmitter to the "slave" inputs of the other connected devices. For more information about an RS-422 interface circuit and a related figure, see the "Using the Inputs with RS-422" section.
Be aware that the last receiver in an RS-422 bus must have a 120 Ohm termination resistor.

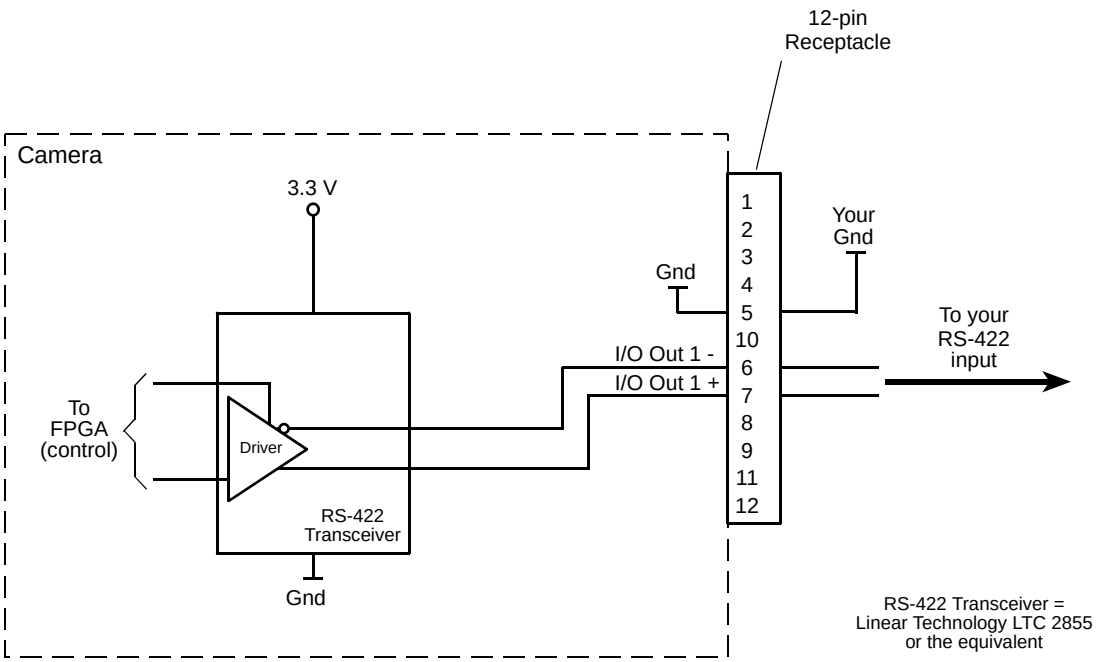



Fig. 17: RS-422 Output Signal

## Using the Outputs with RS-644 LVDS

You cannot directly use the RS-422 signal from a camera output line as an input to an RS-644 low voltage differential signal (LVDS) receiver. However, if a resistor network is placed on the camera's output as shown in Fig. 18, you can use the signal from the camera's output line as an input to an RS-644 device.

For the camera's I/O circuitry to operate properly, you must supply a ground as shown in Fig. 18.

| | |
|---|---|
| | Although the RS-644 standard allows several devices to be connected together in a "multidrop" configuration, we strongly recommend that you do not include any camera output in an RS-644 multidrop. Instead, we strongly recommend that you use a direct, point-to-point connection between the camera and your RS-644 LVDS receiver as shown Fig. 18. |

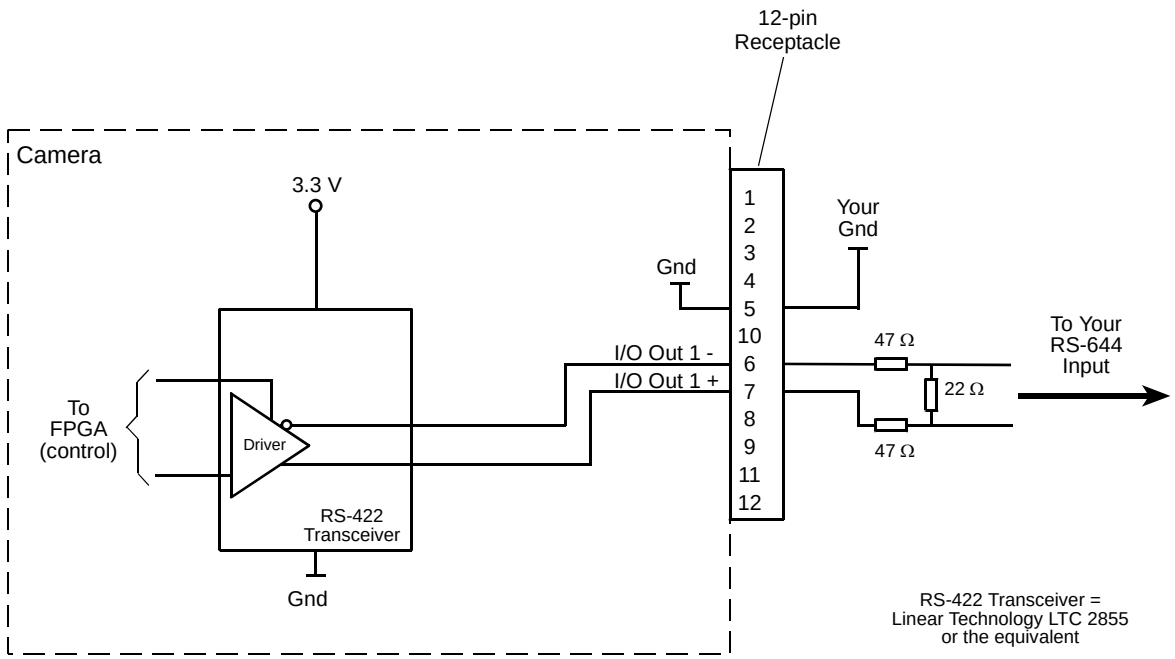



Fig. 18: RS-422 Output Signal Modified for Use with an RS-644 Input

## Using the Outputs with LVTTL

You can use a camera output line as an input to a low voltage TTL receiver, but only if the camera's output signal is used as shown in Fig. 19. In this situation, a low will be indicated by a camera output voltage near zero, and a high will be indicated by a camera output voltage of approximately 3.3 VDC. These voltages are within the typically specified levels for low voltage TTL devices.

For the camera's I/O circuitry to operate properly, you must supply a ground as shown in Fig. 19.

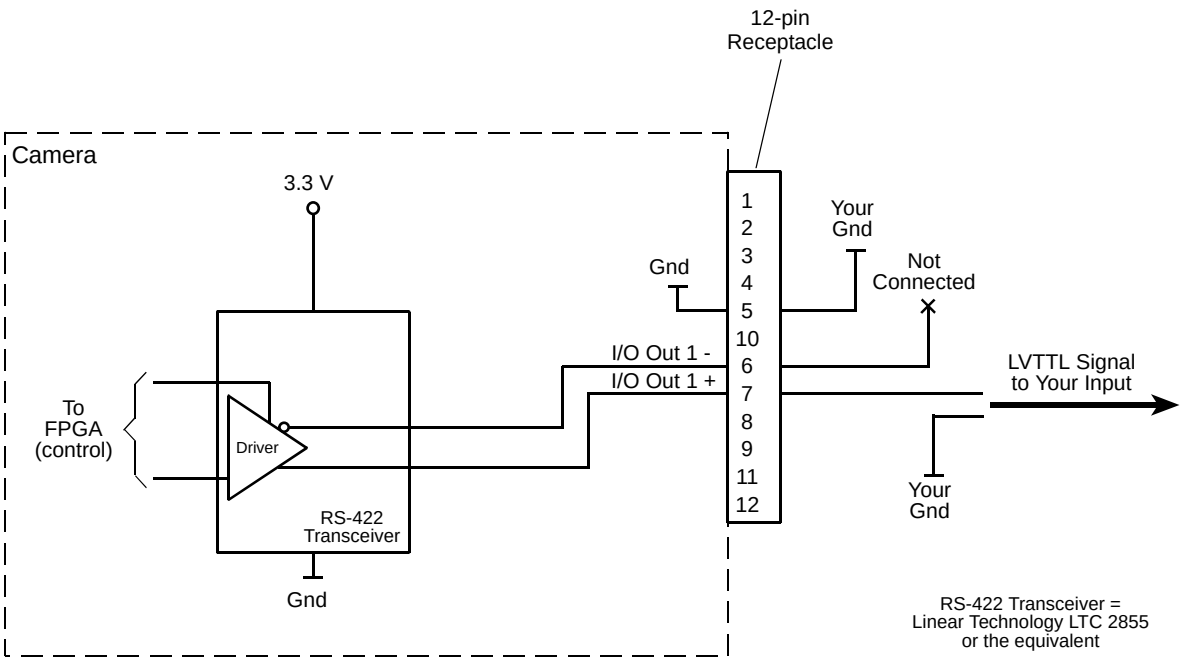



Fig. 19: Output Line Wired for Use with an LVTTL Input

### 7.6.2.2 Minimum Output Pulse Width

You can use the minimum output pulse width feature to ensure that even very narrow camera output signals, e.g. signals originating from a shaft encoder, will reliably be detected by other devices. The MinOutPulseWidthAbs parameter sets output signals for the selected output line to a minimum width. The parameter is set in microseconds and can be set in a range from 0 to 100 µs

You can set the Line Selector and the value of the MinOutPulseWidthAbs parameter from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Select the output line
Camera.LineSelector.SetValue(LineSelector_Out1);

// Set the parameter value to 10.0 microseconds
Camera.MinOutPulseWidthAbs.SetValue(10.0);
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about selecting the source signal for an output line on the camera, see Section 7.7.2.3 on page 65.

For more information about the electrical characteristics of the camera's output lines, see Section 7.7.2 on page 62.

### 7.6.2.3 Output Line Inverters

You can set each individual output line to invert or not to invert the outgoing signal. To set the invert function on an output line:

- Use the Line Selector to select an output line.
- Set the value of the Line Inverter parameter to true to enable inversion on the selected line and to false to disable inversion.

You can set the Line Selector and the Line Inverter parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Enable the inverter on output line 1
Camera.LineSelector.SetValue( LineSelector_Out1 );
Camera.LineInverter.SetValue( true );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

### 7.6.2.4 Selecting the Source Signal for an Output Line

To make a physical output line useful, you must select a source signal for the output line.

The camera has the following standard output signals available that can be selected as the source signal for an output line:

- the Exposure Active signal
- the Acquisition Trigger Wait signal
- the Frame Trigger Wait signal
- the Line Trigger Wait signal

You can also select one of the following as the source signal for an output:

- the "User Output" signal (when you select "user output" as the source signal for an output line, you can use the camera's API to set the state of the line as you desire)
- Off (when "off" is selected as the source signal, the output is disabled.)

To select one of the camera's standard output signals as the source signal for an output line or to select user output or off:

- Use the Line Selector to select an output line.
- Set the value of the Line Source Parameter to Exposure Active, Acquisition Trigger Wait, Frame Trigger Wait, Line Trigger Wait, User Output, or Off. This will select the source signal for the line.

You can set the Line Selector and the Line Source parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Disable output line 1
Camera.LineSelector.SetValue( LineSelector_Out1 );
Camera.LineSource.SetValue( LineSource_Off );

// Select the exposure active signal for output line 1
Camera.LineSelector.SetValue( LineSelector_Out1 );
Camera.LineSource.SetValue( LineSource_ExposureActive );

// Select the acquisition trigger wait for output line 2
Camera.LineSelector.SetValue( LineSelector_Out2 );
Camera.LineSource.SetValue( LineSource_AcquisitionTriggerWait );

// Select the frame trigger wait for output line 2
Camera.LineSelector.SetValue( LineSelector_Out2 );
Camera.LineSource.SetValue( LineSource_FrameTriggerWait );

// Select the line trigger wait signal for output line 2
Camera.LineSelector.SetValue( LineSelector_Out2 );
Camera.LineSource.SetValue( LineSource_LineTriggerWait );
```

```
// Select output line 1 as a user output
Camera.LineSelector.SetValue( LineSelector_Out1 );
Camera.LineSource.SetValue( LineSource_UserOutput );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

For more information about the Exposure Active signal, see Section 8.5.1 on page 124.

For more information about the Acquisition Trigger Wait signal, see Section 8.5.3 on page 126.

For more information about the Frame Trigger Wait signal, see Section 8.5.4 on page 126.

For more information about the Line Trigger Wait signal, see Section 8.5.5 on page 126.

For more information about working with outputs that have "user settable" as the signal source, see Section 7.6.2.5.

### Default Output Line Source Signal Selections

By default, the camera's Exposure Active signal is selected as the source signal for Output Line 1, and the camera's Frame Trigger Wait signal is selected as the source signal for Output Line 2.

## 7.6.2.5 Setting the State of User Settable Output Lines

As mentioned in the previous section, you can select "user output" as the signal source for an output line. For an output line that has "user output" as the signal source, you can use camera parameters to set the state of the line.

### Setting the State of a Single User Output Line

To set the state of a single user output line:

- Use the User Output Selector to select the output line you want to set. For example, if you have designated output line 2 as a user output, you would select output line 2.
- Set the value of the User Output Value parameter to true (high) or false (low). This will set the state of the selected line.

You can set the Output Selector and the User Output Value parameter from within your application software by using the pylon API. The following code snippet illustrates using the API to select "user settable" as the source signal for output line 2 and how to set the state of the output line:

```
// Select "user output" as output line 2 signal source
Camera.LineSelector.SetValue( LineSelector_Out2 );
Camera.LineSource.SetValue( LineSource_UserOutput );
```

```
//Set the state of output line 2 and then read the state
Camera.UserOutputSelector.SetValue( UserOutputSelector_UserOutput2 );
Camera.UserOutputValue.SetValue( true );
bool currentUserOutput2State = Camera.UserOutputValue.GetValue( );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

## Setting the State of Multiple User Output Lines

If you have designated both of the cameras output lines as user outputs, you can use the User Output Value All parameter to set the state of both outputs.

The User Output Value All parameter is a 32 bit value. As shown in Fig. 20, the lowest two bits of the parameter value will set the state of the user outputs. If a bit is 0, it will set the state of the associated output to low. If a bit is high, it will set the state of the associated port to high.

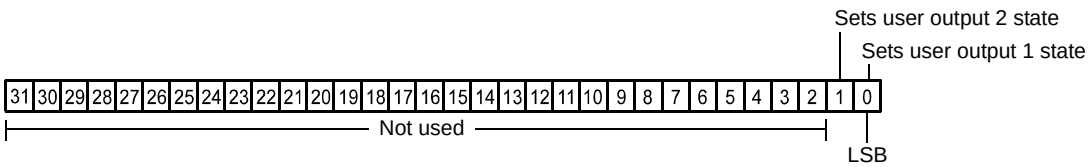



Fig. 20: User Output Value All Parameter Bits

To set the state of multiple user output lines:

- Use the User Output Value All parameter to set the state of multiple user outputs.

You can set the User Output Value All parameter from within your application software by using the pylon API. The following code snippet illustrates using the API to set the parameter:

```
// Set the state of both output lines to 1 and read the state
Camera.UserOutputValueAll.SetValue( 0x3 );
int64_t currentOutputState = Camera.UserOutputValueAll.GetValue( );
```

If you have the invert function enabled on an output line that is designated as a user output, the user setting sets the state of the line before the inverter.

# 7.6.3 Checking the State of the I/O Lines

## Checking the State of All I/O Lines

You can determine the current state of all input and output lines with a single operation. To check the state of all lines:

- Read the value of the Line Status All parameter.

You can read the Line Status All parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to read the parameter value:

```
// Read the line status all value
int64_t lineState = Camera.LineStatusAll.GetValue( );
```

The Line Status All parameter is a 32 bit value. As shown in Fig. 21, certain bits in the value are associated with each line and the bits will indicate the state of the lines. If a bit is 0, it indicates that the state of the associated line is currently low. If a bit is 1, it indicates that the state of the associated line is currently high.

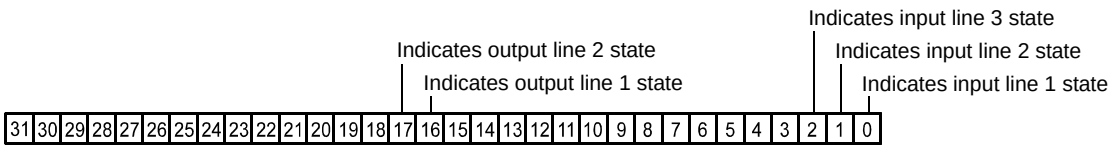

Indicates input line 3 state
Indicates output line 2 state
Indicates input line 2 state
Indicates output line 1 state
Indicates input line 1 state
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Fig. 21: Line Status All Parameter Bits

## Checking the State of a Single Output Line

You can determine the current state of an individual output line. To check the state of a line:

- Use the Line Selector parameter to select an output line.
- Read the value of the Line Status parameter to determine the current state of the selected line. A value of true means the line's state is currently high and a value of false means the line's state is currently low.

You can set the Line Selector and read the Line Status parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and read the parameter value:

```
// Select output line 2 and read the state
Camera.LineSelector.SetValue( LineSelector_Out2 );
bool outputLine2State = Camera.LineStatus.GetValue( );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

## 7.6.4 Checking the Line Logic

### Checking the Line Logic Using Basler pylon

You can determine the type of line logic for each I/O line using Basler pylon:

- Use the Line Selector parameter to select a line.
- Read the value of the Line Logic parameter to determine the type of line logic used by the line. The parameter will indicate whether the logic is positive or negative.

You can set the Line Selector and read the Line Logic parameter value from within your application software by using the Basler pylon API. The following code snippet illustrates using the API to set the selector and read the parameter value:

```
// Select the I/O line and read the line logic type
Camera.LineSelector.SetValue( LineSelector_Line1 );
LineLogicEnums lineLogicLine1 = Camera.LineLogic.GetValue( );
```

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon API and the pylon Viewer, see Section 3.1 on page 17.

## 7.6.5 I/O Line Response Times

In general, the response characteristics for the I/O lines on the camera are as follows:

- Propagation delay for an input receiver (input pins on the camera to the camera's FPGA) is less than 70 ns.
- Propagation delay for an output driver (camera FPGA to the output pins on the camera) is less than 20 ns.
- Signal rise time and signal fall time for the output driver is less than 12.5 ns.

As shown in the I/O schematic at the beginning of this section, the camera's I/O circuitry will incorporate Linear Technology LTC2855 transceivers or the equivalent. For more detailed information about response characteristics, refer to the LTC2855 data sheet.

The response times for the output lines on your camera will fall into the ranges specified above. The exact response time for your specific application will depend on your circuit design.

# 7.7 Ethernet GigE Device Information

The camera uses a standard Ethernet GigE transceiver. The transceiver is fully 100/1000 Base-T 802.3 compliant.

# 8 Acquisition Control

This section provides detailed information about controlling the acquisition of image information. You will find details about triggering frame and line acquisition, about setting the exposure time for acquired lines, about setting the camera's line acquisition rate, and about how the camera's maximum allowed line acquisition rate can vary depending on the current camera settings.

## 8.1 Defining a Frame

As with any other line scan camera, the sensor in a Gigabit Ethernet (GigE) camera is used to perform a series of line acquisitions as an object passes the camera. But unlike many other cameras, GigE line scan cameras do not transmit the pixel data from each individual line to a host PC immediately after the line acquisition is complete. Instead, GigE cameras accumulate acquired lines in a buffer and assembles them into a "frame". When enough line acquisitions have been accumulated to constitute a complete frame, the frame is transmitted via an Ethernet network to a host PC. An acquired frame, therefore, represents a single complete image acquired by the camera.

Three camera parameters, X Offset, Width, and Height are used to define what will constitute a frame.

The X Offset and Width parameters determine which pixels in the sensor line will be used for each line acquisition. The X Offset determines the first pixel to be used and the Width determines the number of pixels to be used. The pixels in the sensor are numbered starting with 0.

Assume, for example, that you are working with a camera that has a 2048 pixel sensor line, that the X Offset parameter is set to 0, and that the Width parameter is set to 2048. In this case, the full length of the sensor line would be used for each line acquisition.

As another example, assume that the X Offset parameter is set to 10 and the Width parameter is set to 25. With these settings, pixels 10 through 34 would be used for each line acquisition as shown in Fig. 22.

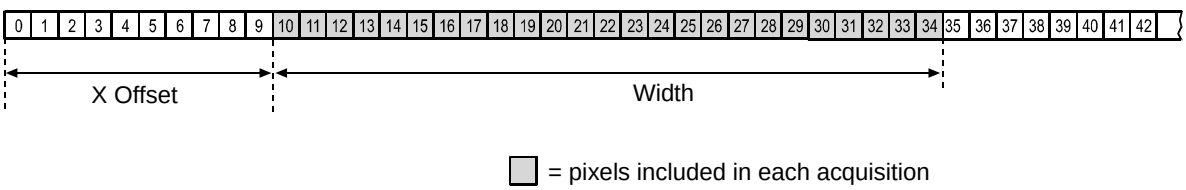



Fig. 22: Pixels Used for Each Line Acquisition

The Height parameter determines the number of lines that will be included in each frame. For example, assume that the Height parameter is set to 100 and that the camera has just started to

acquire lines. In this case, the camera will accumulate acquired line data in an internal buffer until 100 lines have been accumulated. Once pixel data for 100 lines has accumulated in the buffer, the camera will recognize this as a complete frame and it will begin to transmit the acquired frame to your host PC via the GigE network connection. Note that the camera has multiple frame buffers, so it can begin to acquire lines for a new frame as it is transmitting data for the previously acquired frame.

The absolute maximum for the Height parameter value is 4095. Accordingly, a single frame may include 4095 lines at most.

This maximum number of lines can, however, not be obtained under all conditions: In the event of limitations due to the current camera parameter settings or due to the transport layer, the camera will automatically decrease the Height parameter to a suitable value. Each frame will then include fewer lines than originally set.

Given the current camera parameter settings, check the Height parameter to see, whether the desired number of lines per frame can actually be obtained.

## Guidelines When Setting the Frame Parameters

When setting the frame parameters, the following guidelines must be followed:

- The sum of the X Offset parameter plus the Width parameter must be less than or equal to the total number of pixels in the camera's sensor line. For example, if you are working with a camera that has a line with 2048 pixels, the sum of the X Offset setting plus the Width setting must be less than or equal to 2048.
- The Height parameter must be set below the maximum allowed.

  The maximum allowed value for the Height parameter setting will be at least 512, but will vary depending on your camera model and on how the camera's parameters are set. The actual maximum could be considerably greater than 512. To determine the maximum allowed Height value given your current camera settings:

  1. Set all camera parameters other than the Height to your desired values.
  2. Use the technique described in the code snippet on the next page to determine the maximum allowed Height parameter. The value that you retrieve with this technique will give you the maximum allowed Height with all the other current parameter settings taken into account.
  3. Set the Height parameter to a value that is less than or equal to the allowed maximum.

The Width and Height parameters cannot be changed while the camera is in the process of acquiring frames. If the camera receives commands to change the Width or Height parameter values while it is in the process of acquiring frames:

- If the camera is set for single frame mode, the parameters will not change until the current frame is complete or you issue an acquisition stop command.
- If the camera is set for continuous frame mode, the parameters will not change until you issue an acquisition stop command.

## Setting the Frame Parameters

You can set the X Offset, Width, and Height parameter values from within your application software by using the pylon API. The following code snippets illustrate using the API to get the maximum allowed settings and the increments for the Width and Height parameters. They also illustrate setting the X Offset, Width, and Height parameter values

```
int64_t widthMax = Camera.Width.GetMax( );
int64_t widthInc = Camera.Width.GetInc( );
Camera.Width.SetValue( 200 );
Camera.OffsetX.SetValue( 100 );

int64_t heightMax = Camera.Height.GetMax( );
int64_t heightInc = Camera.Height.GetInc( );
Camera.Height.SetValue( 200 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

# 8.2 Controlling Acquisition

Five major elements are involved in controlling the acquisition of images:

- Acquisition start and acquisition stop commands
- The acquisition mode parameter
- Acquisition start triggering
- Frame start triggering
- Line start triggering

## 8.2.1 Acquisition Start and Stop Commands and the Acquisition Mode

The use of Acquisition Start and Acquisition Stop commands and the camera's Acquisition Mode parameter setting are related.

Issuing an Acquisition Start command to the camera prepares the camera to acquire frames. You must issue an Acquisition Start command to the camera before you can begin acquiring frames.

Issuing an Acquisition Stop command to the camera terminates the camera's ability to acquire frames. When the camera receives an Acquisition Stop command:

- If the camera is not in the process of acquiring a frame, its ability to acquire frames will be terminated immediately.
- If the camera is in the process of acquiring a line for a frame, the line acquisition process will be allowed to finish. Frame acquisition will then be stopped, a partial frame will be transmitted, and the camera's ability to acquire frames will be terminated.

The camera's Acquisition Mode parameter has two settings: single frame and continuous.

If the camera's Acquisition Mode parameter is set for single frame, after an Acquisition Start command has been issued to the camera, a single frame can be acquired. When acquisition of one frame is complete, the camera will internally issue an Acquisition Stop command and can no longer acquire frames. To acquire another frame, you must issue a new Acquisition Start command.

If the camera's Acquisition Mode parameter is set for continuous frame, after an Acquisition Start command has been issued to the camera, frame acquisition can be triggered as desired. Each time the proper frame and line triggers are applied, the camera will acquire and transmit a frame. The camera will retain the ability to acquire frames until an Acquisition Stop command has been issued to the camera. Once the Acquisition Stop command is received, the camera can no longer acquire frames.

To see graphical representations of the use of the Acquisition Start and Acquisition Stop commands and the Acquisition Mode parameter, refer to the use case diagrams in Section 8.2.6 on page 86.

### Setting the Acquisition Mode and Issuing Start/Stop Commands

You can set the Acquisition Mode parameter value and you can issue Acquisition Start or Acquisition Stop commands from within your application software by using the pylon API. The code snippet below illustrates using the API to set the Acquisition Mode parameter value and to issue an Acquisition Start command. Note that the snippet also illustrates setting several parameters regarding frame and line triggering. These parameters are discussed later in this chapter.

```
Camera.AcquisitionMode.SetValue( AcquisitionMode_SingleFrame );
Camera.TriggerSelector.SetValue( TriggerSelector_FrameStart );
Camera.TriggerMode.SetValue( TriggerMode_On );
Camera.TriggerActivation.SetValue( TriggerActivation_RisingEdge );
Camera.TriggerSelector.SetValue( TriggerSelector_LineStart );
Camera.TriggerMode.SetValue( TriggerMode_On );
Camera.TriggerActivation.SetValue( TriggerActivation_RisingEdge );
Camera.ExposureMode.SetValue( ExposureMode_Timed );
Camera.ExposureTimeAbs.SetValue( 55 );
Camera.AcquisitionStart.Execute( );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

When the camera's acquisition mode is set to single frame, the maximum possible acquisition frame rate for a given AOI cannot be achieved. This is true because the camera performs a complete internal setup cycle for each single frame.

## 8.2.2 Acquisition Start Triggering

The acquisition start trigger is used in conjunction with the frame start trigger to control the acquisition of frames. In essence, the acquisition start trigger is used as an enabler for the frame start trigger.

When the acquisition start trigger is enabled, the camera’s initial acquisition status is "waiting for acquisition start trigger". When the camera is in this acquisition status, it will ignore any frame start trigger signals it receives. If an acquisition start trigger signal is applied to the camera, it will exit the "waiting for acquisition start trigger" acquisition status and enter the "waiting for frame start trigger" acquisition status. If a frame start trigger signal is applied to the camera, it will exit the "waiting for frame start trigger" acquisition status and enter the "waiting for line start trigger" acquisition status.

In this acquisition status, the camera can react to line start trigger signals and will begin to expose a line each time a proper line start trigger signal is applied.

A primary feature of the acquisition start trigger is that after an acquisition start trigger signal has been applied to the camera and the camera has entered the "waiting for frame start trigger" acquisition status, the camera will return to the "waiting for acquisition start trigger" acquisition status once a specified number of frame start triggers has been received. Before more frames can be acquired, a new acquisition start trigger signal must be applied to the camera to exit it from "waiting for acquisition start trigger". This feature is explained in greater detail in the following sections.

There are two main parameters associated with the acquisition start trigger: the Acquisition Start Trigger Mode parameter and the Acquisition Frame Count parameter.

### 8.2.2.1 Acquisition Start Trigger Mode = Off

When the Trigger Mode parameter for the acquisition start trigger is set to off, the camera will generate all required acquisition start trigger signals internally, and you do not need to apply acquisition start trigger signals to the camera.

### 8.2.2.2 Acquisition Start Trigger Mode = On

When the Acquisition Start Trigger Mode parameter is set to on, you must apply an acquisition start trigger to the camera in order to make the camera’s acquisition state valid. Once an acquisition start trigger has been applied to the camera and the acquisition state has become valid, the state will remain valid until the camera has acquired the number of frames specified by the Acquisition Frame Count parameter. At that point, the acquisition state will become invalid, and you must apply a new acquisition start trigger to the camera before it can acquire any more frames.

When the Acquisition Start Trigger Mode parameter is set to on, you must select a source signal to serve as the acquisition start trigger. The Acquisition Start Trigger Source parameter specifies the source signal. The available selections for the Acquisition Start Trigger Source parameter are:

- Software - When the acquisition start trigger source is set to software, the user applies an acquisition start trigger to the camera by issuing an acquisition start TriggerSoftware command to the camera from the host PC.
- Line 1, line 2 or line 3 - When the acquisition start trigger source is set to line 1, line 2 or line 3, the user applies an acquisition start trigger to the camera by injecting an externally generated acquisition start trigger signal (referred to as an ExASTrig signal) into physical input line 1, line 2 or line 3 on the camera.

If the Acquisition Start Trigger Source parameter is set to Line 1, Line 2 or Line 3, the user must also set the Acquisition Start Trigger Activation parameter. The available settings for the Acquisition Start Trigger Activation parameter are:

- Rising Edge - specifies that a rising edge of the hardware trigger signal will act as the acquisition start trigger.
- Falling Edge - specifies that a falling edge of the hardware trigger signal will act as the acquisition start trigger.

When the Acquisition Start Trigger Mode parameter is set to on, the camera's Acquisition Mode parameter must be set to continuous.

### 8.2.2.3 Acquisition Frame Count

When the Trigger Mode parameter for the acquisition start trigger is set to on, you must set the value of the camera's Acquisition Frame Count parameter. The value of the Acquisition Frame Count can range from 1 to 65535.

With acquisition start triggering on, the camera will initially be in a "waiting for acquisition start trigger" acquisition status. When in this acquisition status, the camera cannot react to frame start trigger signals. If an acquisition start trigger signal is applied to the camera, the camera will exit the "waiting for acquisition start trigger" acquisition status and will enter the "waiting for frame start trigger" acquisition status. It can then react to frame start trigger signals. When the camera has received a number of frame start trigger signals equal to the current Acquisition Frame Count parameter setting, it will return to the "waiting for acquisition start trigger" acquisition status. At that point, you must apply a new acquisition start trigger signal to exit the camera from the "waiting for acquisition start trigger" acquisition status.

### 8.2.2.4 Setting The Acquisition Start Trigger Mode and Related Parameters

You can set the Trigger Mode and Trigger Source parameter values for the acquisition start trigger and the Acquisition Frame Count parameter value from within your application software by using the pylon API.

The following code snippet illustrates using the API to set the acquisition start Trigger Mode to on, the Trigger Source to software, and the Acquisition Frame Count to 5:

```
// Select the acquisition start trigger
Camera.TriggerSelector.SetValue( TriggerSelector_AcquisitionStart );
// Set the mode for the selected trigger
Camera.TriggerMode.SetValue( TriggerMode_On );
// Set the source for the selected trigger
Camera.TriggerSource.SetValue ( TriggerSource_Software );
// Set the acquisition frame count
Camera.AcquisitionFrameCount.SetValue( 5 );
```

The following code snippet illustrates using the API to set the Trigger Mode to on, the Trigger Source to line 1, the Trigger Activation to rising edge, and the Acquisition Frame Count to 5:

```
// Select the acquisition start trigger
Camera.TriggerSelector.SetValue( TriggerSelector_AcquisitionStart );
```

```
// Set the mode for the selected trigger
Camera.TriggerMode.SetValue( TriggerMode_On );
// Set the source for the selected trigger
Camera.TriggerSource.SetValue ( TriggerSource_Line1 );
// Set the activation mode for the selected trigger to rising edge
Camera.TriggerActivation.SetValue( TriggerActivation_RisingEdge );
// Set the acquisition frame count
Camera.AcquisitionFrameCount.SetValue( 5 );
```

You can also use the Basler pylon Viewer application to easily set the parameters.

## 8.2.3 Frame Start Triggering

The frame start trigger is used in conjunction with the line start trigger to control the acquisition of the lines that will be included in each frame. In essence, the frame start trigger is an enabler for the line start trigger, i.e., the camera will only react to line start triggers when the frame start trigger is valid. When the frame start trigger is not valid, line start triggers will be ignored by the camera and will not result in a line acquisition.

The first parameter associated with the frame start trigger is the Trigger Mode parameter. The Trigger Mode parameter has two available settings: off and on.

### 8.2.3.1 Frame Start Trigger Mode = Off

When the Frame Start Trigger Mode parameter is set to off, selection of a source signal for the frame start trigger is not required. With the mode set to off, the camera operates the frame start trigger automatically. How the camera will operate the frame start trigger depends on the setting of the camera's Acquisition Mode parameter:

- If the Acquisition Mode parameter is set to single frame, the camera will automatically make the frame start trigger valid when it receives an Acquisition Start command. The trigger will remain valid until enough lines have been acquired to constitute a complete frame and then will become invalid.
- If the Acquisition Mode parameter is set to continuous frame:
  a. The camera will automatically make the frame start trigger valid when it receives an Acquisition Start command.
  b. The frame start trigger will be held valid until enough lines have been acquired to constitute a complete frame and then will become invalid.
  c. As soon as acquisition of lines for a next frame can start, the frame start trigger will automatically be made valid, will be held valid until enough lines have been acquired to constitute a complete frame, and then will become invalid.
  d. The behavior in step c will repeat until the camera receives an Acquisition Stop command. When an Acquisition Stop command is received, the frame start trigger will become continuously invalid.

### 8.2.3.2 Frame Start Trigger Mode = On

When the Frame Start Trigger Mode parameter is set to on, you must select a source signal for the frame start trigger. The Frame Start Trigger Source parameter specifies the source of the signal. The available selections for the Frame Start Trigger Source parameter are:

- Software - When the frame start trigger source is set to software, the user triggers frame start by issuing a TriggerSoftware command to the camera from the host PC. Each time a TriggerSoftware command is received by the camera, the frame start trigger will become valid and will remain valid until enough lines have been acquired to constitute a complete frame. The frame start trigger will then become invalid
- Line 1 - When the frame start trigger source is set to line 1, the user triggers frame start by applying an external electrical signal (referred to as an ExFSTrig signal) to physical input line 1 on the camera.
- Line 2 - When the frame start trigger source is set to line 2, the user triggers frame start by applying an ExFSTrig signal to physical input line 2 on the camera.
- Line 3 - When the frame start trigger source is set to line 3, the user triggers frame start by applying an ExFSTrig signal to physical input line 3 on the camera.
- Shaft Encoder Module Out - When the frame start trigger source is set to shaft encoder module out, the output signal from the camera's shaft encoder software module will trigger frame start.

If the Frame Start Trigger Source parameter is set to Line 1, Line 2, Line 3, or Shaft Encoder Module Out, the user must also set the Frame Start Trigger Activation parameter. The available settings for the Frame Start Trigger Activation parameter are:

- Rising Edge - specifies that a rising edge of the source signal will make the frame start trigger valid. The frame start trigger will remain valid until enough lines have been acquired to constitute a complete frame and then will become invalid.
- Falling Edge - specifies that a falling edge of the source signal will make the frame start trigger valid. The frame start trigger will remain valid until enough lines have been acquired to constitute a complete frame and then will become invalid.
- Level High - specifies that a rising edge of the source signal will make the frame start trigger valid. The frame start trigger will remain valid as long as the signal remains high. The frame start trigger will become invalid when the signal becomes low.
- Level Low - specifies that a falling edge of the source signal will make the frame start trigger valid. The frame start trigger will remain valid as long as the signal remains low. The frame start trigger will become invalid when the signal becomes high.

If the Frame Start Trigger Activation parameter is set to Level High or Level Low, the user must also set the Partial Closing Frame parameter. The available settings for the Partial Closing Frame parameter are:

- True: When the frame start trigger signal transitions while a frame is being acquired frame acquisition will stop and only the portion of the frame acquired so far will be transmitted.
- False - When the frame start trigger signal transitions while a frame is being acquired the complete frame will be acquired and transmitted.

By default, Input Line 3 is selected as the source signal for the Frame Start Trigger.

If the Frame Start Trigger Source parameter is set to Shaft Encoder Module Out, the recommended setting for the Frame Start Trigger Activation parameter is Rising Edge.

If the Frame Start Trigger Source parameter is set to Line 1, Line 2, or Line 3, the electrical signal applied to the selected input line must be held high for at least 100 ns for the camera to detect a transition from low to high and must be held low for at least 100 ns for the camera to detect a transition from high to low.

To see graphical representations of frame start triggering, refer to the use case diagrams in Section 8.2.6 on page 86.

### 8.2.3.3 Setting the Frame Start Trigger Parameters

You can set the Trigger Mode, Trigger Source, and Trigger Activation parameter values for the frame start trigger from within your application software by using the pylon API. If your settings make it necessary, you can also issue a Trigger Software command. The following code snippet illustrates using the API to set the frame start trigger to mode = on, with rising edge triggering on input line 1:

```
// Select the trigger you want to work with
Camera.TriggerSelector.SetValue( TriggerSelector_FrameStart );
// Set the mode for the selected trigger
Camera.TriggerMode.SetValue( TriggerMode_On );
// Set the source for the selected trigger
Camera.TriggerSource.SetValue ( TriggerSource_Line1 );
// Set the activation scheme for the selected trigger
Camera.TriggerActivation.SetValue( TriggerActivation_RisingEdge );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

If you are using a color camera, you have spatial correction enabled, and you have the frame start trigger mode set to off, you must discard the first n x 2 lines from the first frame transmitted by the camera after an acquisition start command is issued (where n is the absolute value of the current spatial correction parameter setting).

If you have spatial correction enabled and you have the frame start trigger mode set to on, you must discard the first n x 2 lines from each frame transmitted by the camera.

For more information about spatial correction and the spatial correction parameter, see Chapter 9 on page 133.

### 8.2.3.4 Frame Timeout

The Frame Timeout allows setting a maximum time (in microseconds) that may elapse for each frame acquisition, i.e. the maximum time for the acquisition of the lines for a frame.

When the frame timeout is enabled and a time is set a partial frame will be transmitted if the set time has elapsed before all lines specified for the frame are acquired. In addition, a frame timeout event will be generated if it was enabled.

You can enable and configure the frame timeout from within your application software by using the pylon API. The following code snippet illustrates using the API to enable and configure the frame timeout:

```
// enable FrameTimeout and set FrameTimeout value
Camera.FrameTimeoutEnable.SetValue(true);
// Although FrameTimeoutAbs is measured in microseconds the current
resolution
// is just milliseconds.
double FrameTimeout_us = 20000.0; // 20 ms
Camera.FrameTimeoutAbs.SetValue(FrameTimeout_us);
```

You can enable the frame timeout event from within your application software by using the pylon API. The following code snippet illustrates using the API to enable the frame timeout event:

```
// enable FrameTimeout event
Camera.EventSelector.SetValue(EventSelector_FrameTimeout);
Camera.EventNotification.SetValue(EventNotification_GenICamEvent);

// In order to capture FrameTimeout events:
// Set up an event grabber and register a callback for
// the node 'FrameTimeoutEventPort'
```

For more information about event reporting and enabling an event, see Section 11.4 on page 188.

## 8.2.4 Line Start Triggering

The line start trigger is used to start a line acquisition. Keep in mind that the camera will only react to a line start trigger when the frame start trigger is valid. If the frame start trigger is invalid, line start triggers will be ignored.

The first parameter associated with the line start trigger is the Trigger Mode parameter. The Trigger Mode parameter has two available settings: off and on.

### 8.2.4.1 Line Start Trigger Mode = Off

When the Line Start Trigger Mode parameter is set to off, selection of a source signal for the line start trigger is not required. With the mode set to off, the camera operates the line start trigger automatically. How the camera will operate the line start trigger depends on the setting of the camera's Acquisition Mode parameter:

- If the Acquisition Mode parameter is set to single frame, the camera will automatically begin generating line start triggers when it receives an Acquisition Start command. The camera will generate line start triggers until enough lines have been acquired to constitute a complete frame and then will stop generating line start triggers.
- If the Acquisition Mode parameter is set to continuous frame, the camera will automatically begin generating line start triggers when it receives an Acquisition Start command. The camera will continue to generate line start triggers until it receives an Acquisition Stop command.

The rate at which the line start triggers are generated will be determined by the camera's Acquisition Line Rate Abs parameter:

- If the parameter is set to a value less than the maximum allowed line acquisition rate, the camera will generate triggers at the rate specified by the parameter setting.
- If the parameter is set to a value greater than the maximum allowed line acquisition rate, the camera will generate line start triggers at the maximum allowed line rate.

For more information about the maximum allowed line rate, see Section 8.7 on page 119.

#### Exposure Time Control with Line Start Trigger Mode Off

When the line start trigger mode is set to off, the exposure time for each line acquisition is determined by the value of the camera's Exposure Time parameters.

For more information about the camera's exposure time parameters, see Section  on page 103.

### 8.2.4.2 Line Start Trigger Mode = On

When the Line Start Trigger Mode parameter is set to on, you must select a source signal for the line start trigger. The Line Start Trigger Source parameter specifies the source signal. The available selections for the Line Start Trigger Source parameter are:

- Software - When the line start trigger source is set to software, the user triggers line start by issuing a TriggerSoftware command to the camera from the host PC. Each time a TriggerSoftware command is received by the camera, the line start trigger will become valid. It will become invalid during line acquisition and will become valid again when the next TriggerSoftware command is received and when the camera is ready again for a new line acquisition.
- Line 1 - When the line start trigger source is set to line 1, the user triggers each line acquisition start by applying an external electrical signal (referred to as an ExLSTrig signal) to physical input line 1 on the camera.
- Line 2 - When the line start trigger source is set to line 2, the user triggers each line acquisition start by applying an ExLSTrig signal to physical input line 2 on the camera.
- Line 3 - When the line start trigger source is set to line 3, the user triggers each line acquisition start by applying an ExLSTrig signal to physical input line 3 on the camera.
- Shaft Encoder Module Out - When the line start trigger source is set to shaft encoder module out, the output signal from the camera's shaft encoder software module will trigger each line acquisition start.

If the Line Start Trigger Source parameter is set to Line 1, Line 2, Line 3, or Shaft Encoder Module Out, the user must also set the Line Start Trigger Activation parameter. The available settings for the Line Start Trigger Activation parameter are:

- Rising Edge - specifies that a rising edge of the source signal will start a line acquisition.
- Falling Edge - specifies that a falling edge of the source signal will start a line acquisition.

By default, Input Line 1 is selected as the source signal for the Line Start Trigger.

All line start trigger signals input into the camera when the frame start trigger signal is invalid will be ignored by the camera.

If the Line Start Trigger Source parameter is set to Shaft Encoder Module Out, the recommended setting for the Line Start Trigger Activation parameter is Rising Edge.

If the Line Start Trigger Source parameter is set to Line 1, Line 2, or Line 3, the electrical signal applied to the selected input line must be held high for at least 100 ns for the camera to detect a transition from low to high and must be held low for at least 100 ns for the camera to detect a transition from high to low..

## Exposure Time Control with Line Start Trigger Mode On

When the Line Start Trigger Mode parameter is set to on, there are three modes available to control the exposure time for each acquired line: trigger width control, timed control, and control off. You can set the camera's Exposure Mode parameter to select one of the exposure time control modes. The modes are explained in detail below.

If you have the Line Start Trigger Source parameter set to Line 1, Line 2, or Line 3, any one of the three exposure control modes will work well. You should select the mode that is most appropriate for your application.

If you have the Line Start Trigger Source parameter set to Shaft Encoder Module out, we recommend that you select either the timed control mode or the control off mode. The trigger width mode should not be used in this case.

In all cases, the exposure time for each line must be within the minimum and the maximum stated in Table 7 on page 83. This is true regardless of the method used to control exposure.

### Trigger Width Exposure Time Control Mode

When the trigger width exposure time control mode is selected, the exposure time for each line acquisition will be directly controlled by the source signal for the line start trigger. If the camera is set for rising edge triggering, the exposure time begins when the signal rises and continues until the signal falls. If the camera is set for falling edge triggering, the exposure time begins when the signal falls and continues until the signal rises. Fig. 23 illustrates trigger width exposure with the camera set for rising edge line start triggering.

Trigger width exposure is especially useful if you intend to vary the length of the exposure time for each acquired line.

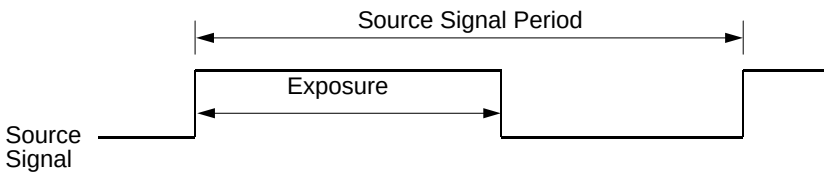



Fig. 23: Trigger Width Exposure with Rising Edge Line Start Triggering

### Timed Exposure Control Mode

When the timed exposure control mode is selected, the exposure time for each line acquisition is determined by the value of the camera's Exposure Time parameters. If the camera is set for rising edge triggering, the exposure time starts when the source signal for the line start trigger rises. If the camera is set for falling edge triggering, the exposure time starts when the source signal falls. Fig. 24 illustrates timed exposure with the camera set for rising edge line start triggering.

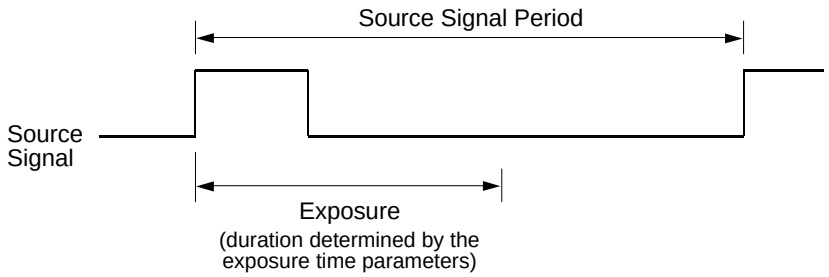



Fig. 24: Timed Exposure with Rising Edge Line Start Triggering

For more information about the camera's exposure time parameters, see Section 8.2.5.2 on page 84.

### Exposure Time Control Mode Off

When the exposure control mode is set to off, each acquired line will be exposed for the full period of the source signal for the line start trigger. This will be true regardless of whether the camera is set for rising edge or for falling edge triggering. Fig. 25 illustrates exposure with the exposure mode set to off.

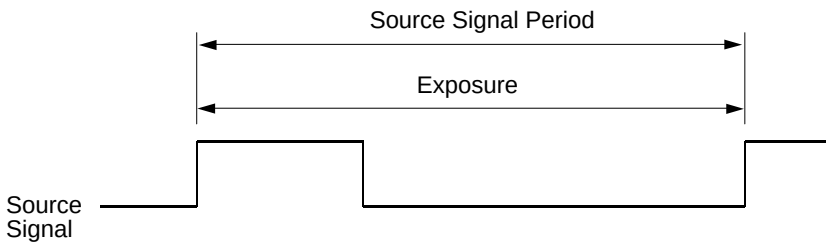



Fig. 25: Exposure with the Mode Set to Off

## Exposure Start and Exposure End Delays

When the line start trigger mode is set to on and an input line is selected as the source signal for the line start trigger, there is a delay between the transition of the line start signal and the actual start of exposure. For example, if you are using the timed exposure mode with rising edge triggering, there is a delay between the rise of the signal and the actual start of exposure.

There is also an exposure end delay, i.e., a delay between the point when exposure should end as explained in the diagrams on the previous page and when it actually does end.

The base exposure start and end delays are as shown in Table 5:

| | raL2048-48gm | raL4096-24gm |
|---|---|---|
| Start Delay | 1.5 µs | 1.5 µs |
| End Delay | 1.2 µs | 1.2 µs |

Table 5: Base Exposure Start and End Delays

When using the frequency converter, the delay values may slightly differ from those given in Table 5.

There is also a second component to the start and end delays. This second component is the debouncer setting for the input line. The debouncer setting for the input line must be added to the base start and end delays shown in Table 5 to determine the total start delay and end delay.
For example, assume that you are using an raL2048-48gm camera and that you have set the line start trigger mode to on. Also assume that you have selected input line 1 as the source signal for the line start trigger and that the debouncer parameter for line 1 is set to 5 µs. In this case:

Total Start Delay = Start Delay Value from Table 5 + Debouncer Setting

Total Start Delay = 1.5 µs + 5 µs

Total Start Delay = 6.5 µs

Total End Delay = End Delay Value from Table 5 + Debouncer Setting

Total End Delay = 1.2 µs + 5 µs

Total End Delay = 6.2 µs

### 8.2.4.3 Setting the Line Start Trigger Parameters

You can set the Trigger Mode, Trigger Source, and Trigger Activation parameter values for the line start trigger from within your application software by using the pylon API. If your settings make it necessary, you can also select an exposure mode and set the exposure time.

The following code snippet illustrates using the API to set the line start trigger to mode = off, the line rate to 20000, and the exposure time to 50 µs:

```
// Select the trigger you want to work with
Camera.TriggerSelector.SetValue( TriggerSelector_LineStart );
// Set the mode for the selected trigger
Camera.TriggerMode.SetValue( TriggerMode_Off );
// set a line rate
Camera.AcquisitionLineRateAbs.SetValue( 20000 );
// set the exposure time to 50 µs
Camera.ExposureTimeAbs.SetValue( 50.0 );
```

The following code snippet illustrates using the API to set the line start trigger to mode = on, to set rising edge triggering on input line 2, to set the exposure mode to timed, and to set the exposure time to 60 µs:

```
// Select the trigger you want to work with
Camera.TriggerSelector.SetValue( TriggerSelector_LineStart );
// Set the mode for the selected trigger
Camera.TriggerMode.SetValue( TriggerMode_On );
// Set the source for the selected trigger
Camera.TriggerSource.SetValue ( TriggerSource_Line2 );
// Set the activation for the selected trigger
Camera.TriggerActivation.SetValue( TriggerActivation_RisingEdge );

// set for the timed exposure mode and set exposure time to 60 µs
Camera.ExposureMode.SetValue( ExposureMode_Timed );
Camera.ExposureTimeAbs.SetValue( 60.0 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

## 8.2.5 Exposure Time

As described in Section 8.2.4.1 on page 77, when you are operating the camera with the Line Start Trigger Mode set to Off, the exposure time for each line acquisition will be determined by the camera's exposure time parameters.

As described in Section 8.2.4.2 on page 78, when you are operating the camera with the Line Start Trigger Mode set to On, the exposure time for each line acquisition may be controlled by an external signal or it may be determined by the exposure time parameters.

### 8.2.5.1 Minimum and Maximum Exposure Times

If you are operating the camera in either of these two ways:

- the Line Start Trigger Mode is set to Off
- the Line Start Trigger Mode is set to On and the Timed Exposure Time Control Mode is selected

the exposure time will be determined by the settings for the camera's exposure time parameters. The minimum and the maximum allowed exposure time for each acquired line are as shown in Table 6.

| | raL2048-48gm | raL4096-24gm |
|---|---|---|
| Min | 2.0 µs | 2.0 µs |
| Max | 10000 µs | 10000 µs |

Table 6: Minimum and Maximum Allowed Exposure Times

If you are operating the camera in either of these two ways:

- the Line Start Trigger Mode is set to On and the Trigger Width Exposure Time Control Mode is selected
- the Line Start Trigger Mode is set to On and Exposure Time Control Mode Off is selected

The exposure time for each acquired line will be controlled by an external signal. The minimum allowed exposure time for each acquired line is as shown in Table 7 and there is no limit on the maximum exposure time. Keep in mind, however, that using a very long exposure time can lead to significant degradation of the image quality.

| | raL2048-48gm | raL4096-24gm |
|---|---|---|
| Min | 2.0 µs | 2.0 µs |

Table 7: Minimum Allowed Exposure Times

### 8.2.5.2 Exposure Time Parameters

If you are operating the camera in either of the following ways, you must specify an exposure time by setting the camera's exposure time parameters:

- the Line Start Trigger Mode is set to Off
- the Line Start Trigger Mode is set to On and the Timed Exposure Time Control Mode is selected

There are two ways to specify the exposure time: by setting "raw" parameter values or by setting an "absolute" parameter value. The two methods are described below. You can use whichever method you prefer to set the exposure time.

#### Setting the Exposure Time Using "Raw" Settings

When exposure time is set using "raw" values, the exposure time will be determined by a combination of two elements. The first element is the value of the Exposure Time Raw parameter, and the second element is the Exposure Time Base. The exposure time is determined by the product of these two elements:

Exposure Time = (Exposure Time Raw Parameter Value) x (Exposure Time Base Abs Parameter Value)

The Exposure Time Raw parameter value can be set in a range from 1 to 4095.

By default, the Exposure Time Base Abs parameter is set to a value of 1.0 µs on all camera models. The Exposure Time Base Abs parameter can be changed in increments of 0.1 µs.

The product of the Exposure Time Raw parameter setting and the Exposure Time Base Abs parameter setting (i.e., the exposure time) must be equal to or greater than the minimum exposure specified in the table on the previous page. It is possible to use the parameters to set the exposure time lower than what is shown in the table, but this is not allowed and the camera will not operate properly when set this way.

If you are using a GenICam compliant tool such as the Basler pylon Viewer and you attempt to set the exposure time to exactly the minimum allowed or to exactly the maximum allowed, you will see unusual error codes. This is an artifact of a rounding error in the GenICam interface architecture. As a work around, you could set the exposure time slightly above the minimum or below the maximum. Values between the minimum and the maximum are not affected by the problem.

You can set the Exposure Time Raw and Exposure Time Base Abs parameter values from within your application software by using the pylon API. The following code snippet illustrates using the API to set the parameter values:

```
Camera.ExposureMode.SetValue( ExposureMode_Timed );
Camera.ExposureTimeRaw.SetValue( 2 );
Camera.ExposureTimeBaseAbs.SetValue( 25.0 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

### Setting the Exposure Time Using "Absolute" Settings

You can also set the exposure time with an "absolute" parameter. This is accomplished by setting the camera's Exposure Time Abs parameter. The unit for the Exposure Time Abs parameter is µs.

The increment for the Exposure Time Abs parameter is determined by the current setting for the Exposure Time Base Abs parameter. For example, if the time base parameter is currently set to 62.0 µs, you could set the Exposure Time Abs parameter to 62.0 µs, 124.0 µs, 186.0 µs, etc.

Note that if you set the Exposure Time Abs parameter to a value that is not a multiple of the Exposure Time Base parameter, the camera will automatically change the setting for the Exposure Time Abs parameter to the nearest multiple of the time base parameter.

You should also be aware that if you change the exposure time using the raw settings, the Exposure Time Abs parameter will automatically be updated to reflect the new exposure time.

You can set the Exposure Time Abs parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to set the parameter value:

```
Camera.ExposureTimeAbs.SetValue( 124.0 );
double resultingExpTime = Camera.ExposureTimeAbs.GetValue( );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

The setting for the Exposure Time Abs parameter must be between the minimum and the maximum allowed values (inclusive) shown in Table 5 on page 81..

## 8.2.6 Use Case Descriptions and Diagrams

The following pages contain a series of use case descriptions and diagrams. The descriptions and diagrams are designed to illustrate how acquisition start triggering, frame start triggering and line start triggering will work with common combinations of parameter settings.

These use cases do not represent every possible combination of the parameters associated with acquisition start, frame start, and line start triggering. They are simply intended to aid you in developing an initial understanding of how triggers and parameters interact.

In each diagram, the black box in the upper left corner indicates how the parameters are set. Note that the number of Lines Per Frame (Height) parameter, is set to three for each diagram. This is not realistic, but is used in the diagrams so that they will more conveniently fit onto a single page.

### Use Case 1 - Acquisition Start, Frame Start, and Line Start Triggering Off (Free Run), Single Frame Mode

Use case one is illustrated on page 86.

In this use case, the Acquisition Start Trigger Mode, the Frame Start Trigger Mode, and the Line Start Trigger Mode parameters are all set to off. The camera will internally manage acquisition start, frame start, and line start trigger signals. When the camera is set this way, it will acquire lines without any need for triggering by the user. This use case is commonly referred to as "free run".

The rate at which the camera will acquire lines will normally be determined by the camera’s Acquisition Line Rate Abs parameter. If the Acquisition Line Rate Abs parameter is disabled, the camera will acquire lines at the maximum allowed line rate.

In this example, each frame is set to include three lines.

When the Acquisition Mode is set to Single Frame, an acquisition start command must be issued for the acquisition of each single frame.

**Settings:**
Acquisition Mode = Single Frame

Acquisition Start Trigger Mode = Off

Frame Start Trigger Mode = Off
Lines Per Frame (Height) = 3

Line Start Trigger Mode = Off

= trigger signal internally generated by the camera

= camera is waiting for an acquisition start trigger signal

= camera is waiting for a frame start trigger signal

= camera is waiting for a line start trigger signal

= line exposure and readout

= frame transmitted

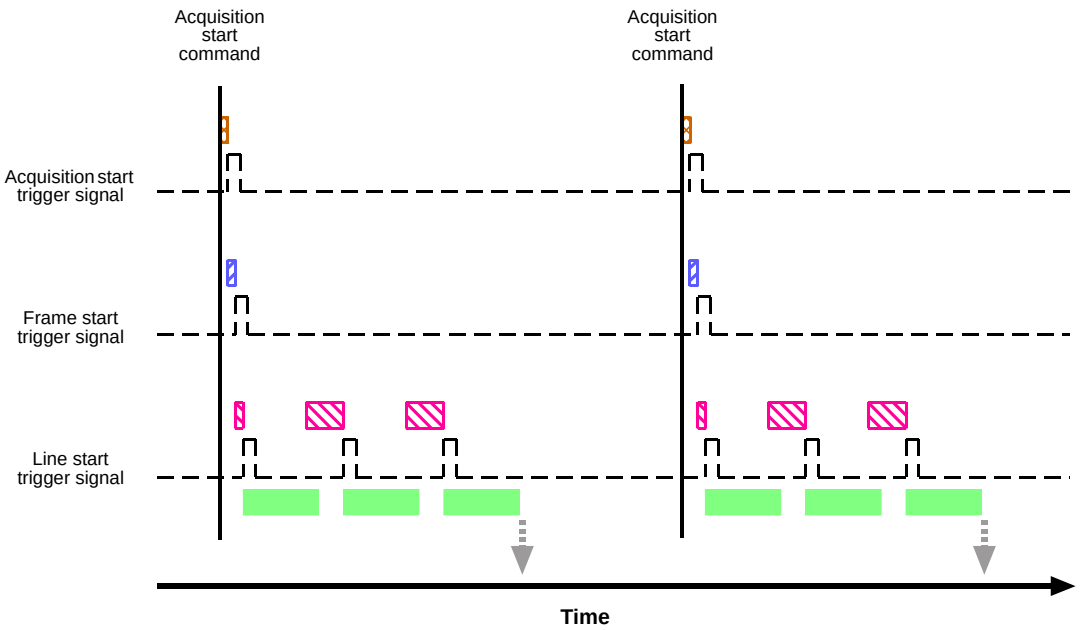



Fig. 26: Use Case Diagram - Single Frame Mode with Acquisition Start, Frame Start, and Line Start Triggering Set to Off

## Use Case 2 - Acquisition Start, Frame Start, and Line Start Triggering Off (Free Run), Continuous Frame Mode

Use case two is illustrated on page 88.

This use case is equivalent to the preceding use case one, except for the fact that the acquisition mode is set to Continuous Frame.

In this use case, the Acquisition Start Trigger Mode, the Frame Start Trigger Mode, and the Line Start Trigger Mode parameters are all set to off. The camera will internally manage acquisition start, frame start, and line start trigger signals. When the camera is set this way, it will constantly acquire lines without any need for triggering by the user. This use case is commonly referred to as "free run".

The rate at which the camera will acquire lines will normally be determined by the camera's Acquisition Line Rate Abs parameter. If the Acquisition Line Rate Abs parameter is disabled, the camera will acquire lines at the maximum allowed line rate.

In this example, each frame is set to include three lines.

When the Acquisition Mode is set to Continuous Frame, the camera will acquire frames until an acquisition stop command is issued.

If an acquisition stop command is issued when not all lines of the current frame are yet acquired, the partial frame will be transmitted.

**Settings:**
Acquisition Mode = Continuous Frame

Acquisition Start Trigger Mode = Off

Frame Start Trigger Mode = Off
Lines Per Frame (Height) = 3

Line Start Trigger Mode = Off

= trigger signal internally generated by the camera

= camera is waiting for an acquisition start trigger signal

= camera is waiting for a frame start trigger signal

= camera is waiting for a line start trigger signal

= line exposure and readout

= complete frame transmitted

= partial frame transmitted

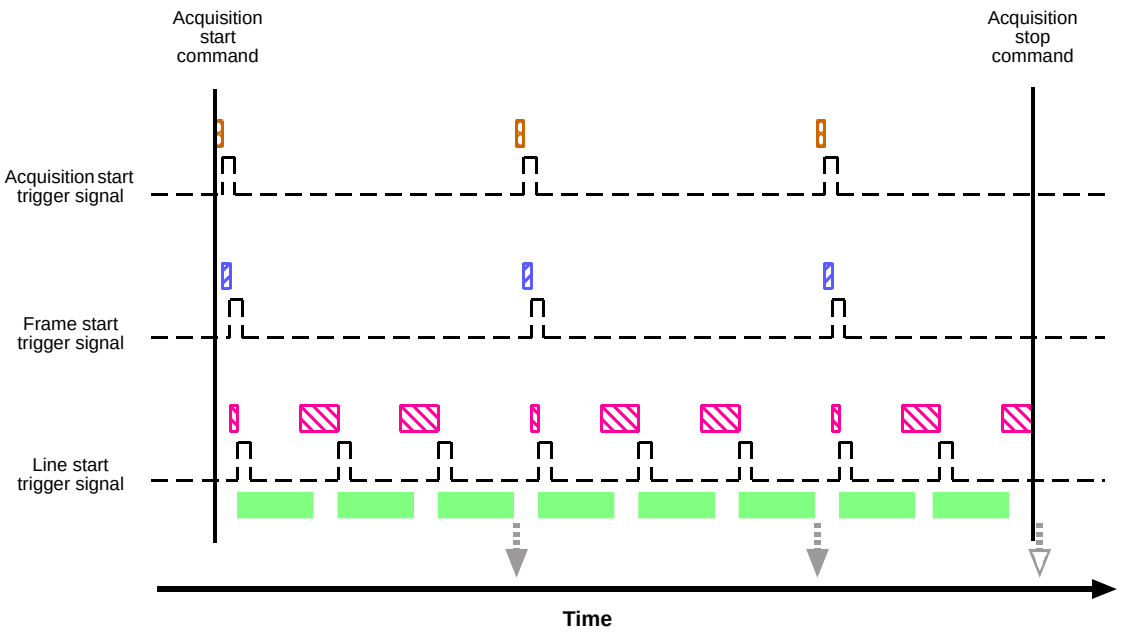



Fig. 27: Use Case Diagram - Continuous Frame Mode with Acquisition Start, Frame Start and Line Start Triggering Set to Off

### Use Case 3 - Acquisition Start and Line Start Triggering Off (Free Run), Frame Start Triggering On

Use case three is illustrated on page 91.

In this use case, the Acquisition Start Trigger Mode and the Line Start Trigger Mode parameters are set to off. The camera will internally manage acquisition start and line start trigger signals without any need for triggering by the user ("free run").

The Frame Start Trigger Mode parameter is set to on, requiring that a frame start trigger signal is applied to the camera.

The rate at which the camera will acquire lines will normally be determined by the camera's Acquisition Line Rate Abs parameter. If the Acquisition Line Rate Abs parameter is disabled, the camera will acquire lines at the maximum allowed line rate. Note that the overall line rate will also depend on the frame start trigger signal: Lines will only be acquired after a related preceding frame start trigger signal has transitioned.

In this example, each frame is set to include three lines.

When the Acquisition Mode is set to Continuous Frame, the camera will be set to acquire frames until an acquisition stop command is issued.

**Settings:**
Acquisition Mode = Continuous Frame

Acquisition Start Trigger Mode = Off

Frame Start Trigger Mode = On
Frame Start Trigger Source = Line 2
Frame Start Trigger Activation = Rising Edge
Lines Per Frame (Height) = 3

Line Start Trigger Mode = Off

= trigger signal internally generated by the camera

= trigger signal applied by the user

= camera is waiting for an acquisition start trigger signal

= camera is waiting for a frame start trigger signal

= camera is waiting for a line start trigger signal

= line exposure and readout

= frame transmitted

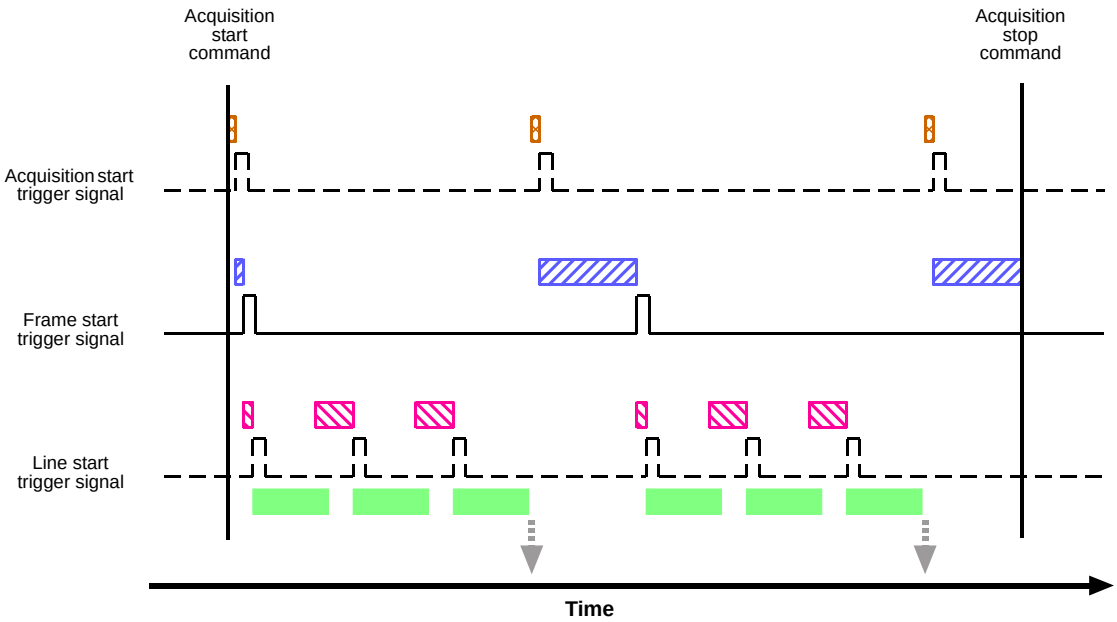



Fig. 28: Use Case Diagram - Continuous Frame Mode with Acquisition Start and Line Start Triggering Set to Off and Frame Start Triggering Set to On

## Use Case 4 - Acquisition Start Triggering Off (Free Run), Frame Start and Line Start Triggering On

Use case four is illustrated on page 92.

In this use case, the Acquisition Start Trigger Mode parameter is set to off. The camera will internally manage acquisition start trigger signals without any need for triggering by the user ("free run").

The Frame Start Trigger Mode and the Line Start Trigger Mode parameters are set to on, requiring that frame start and line start trigger signals are applied to the camera.

The rate at which the camera will acquire lines will be determined by the line start trigger signal and must be below the maximum allowed line rate determined by the current setting. Note that the overall line rate will also depend on the frame start trigger signal: Lines will only be acquired after a related preceding transition of frame start trigger signal has occurred.

In this example, each frame is set to include three lines.

When the Acquisition Mode is set to Continuous Frame, the camera will be set to acquire frames until an acquisition stop command is issued.

**Settings:**
Acquisition Mode = Continuous Frame

Acquisition Start Trigger Mode = Off

Frame Start Trigger Mode = On
Frame Start Trigger Source = Line 2
Frame Start Trigger Activation = Rising Edge
Lines Per Frame (Height) = 3

Line Start Trigger Mode = On
Line Start Trigger Source = Line 3
Line Start Trigger Activation = Rising Edge

= trigger signal internally generated by the camera

= trigger signal applied by the user

= camera is waiting for an acquisition start trigger signal

= camera is waiting for a frame start trigger signal

= camera is waiting for a line start trigger signal

= line exposure and readout

= line start trigger signal is ignored because the camera is waiting for a frame start trigger signal

= frame transmitted

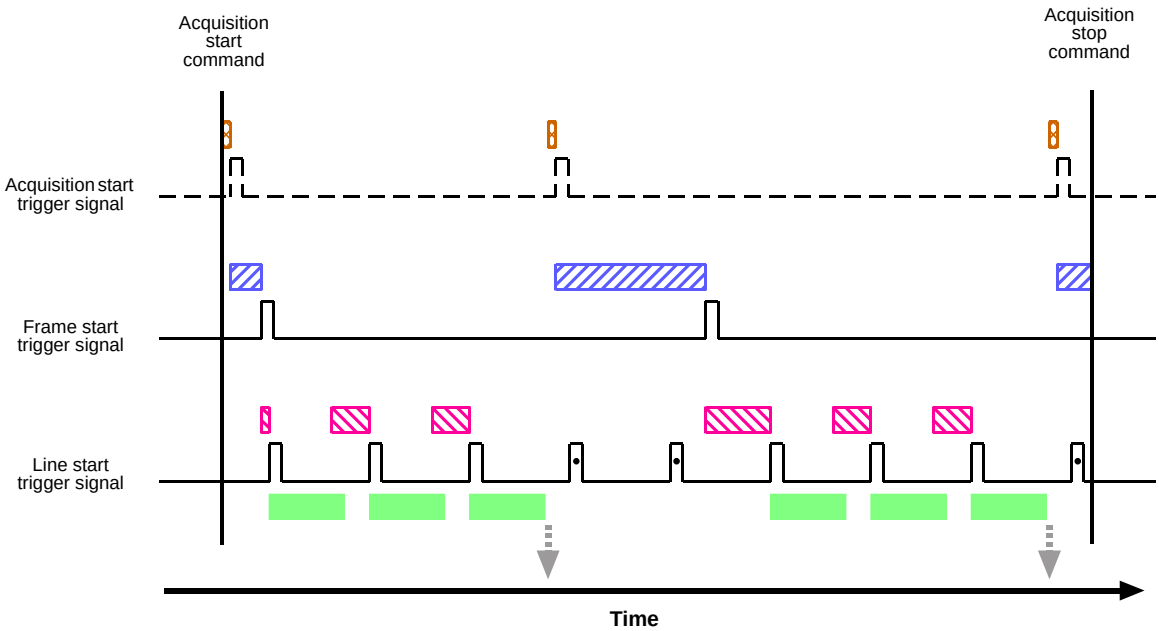



Fig. 29: Use Case Diagram - Continuous Frame Mode with Acquisition Start Triggering Set to Off and Frame Start and Line Start Triggering Set to On

### Use Case 5 - Acquisition Start Triggering Off (Free Run), Frame Start and Line Start Triggering On, Frame Start Trigger Level High, Partial Closing Frame False

Use case five is illustrated on page 94.

In this use case, the Acquisition Start Trigger Mode parameter is set to off. The camera will internally manage acquisition start trigger signals without any need for triggering by the user ("free run").

The Frame Start Trigger Mode and the Line Start Trigger Mode parameters are set to on, requiring that frame start and line start trigger signals are applied to the camera.

The Frame Start Trigger Activation is set to Level High. This means that a transition of the frame start trigger signal is always be present as long as the signal stays high. Accordingly, during this period frames can be acquired without interruption which otherwise will happen if a related preceding transition of the frame start trigger signal has not occurred (c.f. also use case four).

In this example, each frame is set to include three lines.

In this example, the frame start trigger signal goes low while a frame is being acquired (i.e. while line one of the closing frame of the sequence of frames is being acquired). However, with Partial Closing Frame set to false, the complete closing frame will be acquired and transmitted.

When the Acquisition Mode set to Continuous Frame, the camera will be set to acquire frames until an acquisition stop command is issued.

**Settings:**
Acquisition Mode = Continuous Frame

Acquisition Start Trigger Mode = Off

Frame Start Trigger Mode = On
Frame Start Trigger Source = Line 2
Frame Start Trigger Activation = Level High
Partial Closing Frame = False
Lines Per Frame (Height) = 3

Line Start Trigger Mode = On
Line Start Trigger Source = Line 3
Line Start Trigger Activation = Rising Edge

= trigger signal internally generated by the camera

= trigger signal applied by the user

= camera is waiting for an acquisition start trigger signal

= camera is waiting for a frame start trigger signal

= camera is waiting for a line start trigger signal

= line exposure and readout

= line start trigger signal is ignored because the camera is waiting for a frame start trigger signal

= frame transmitted

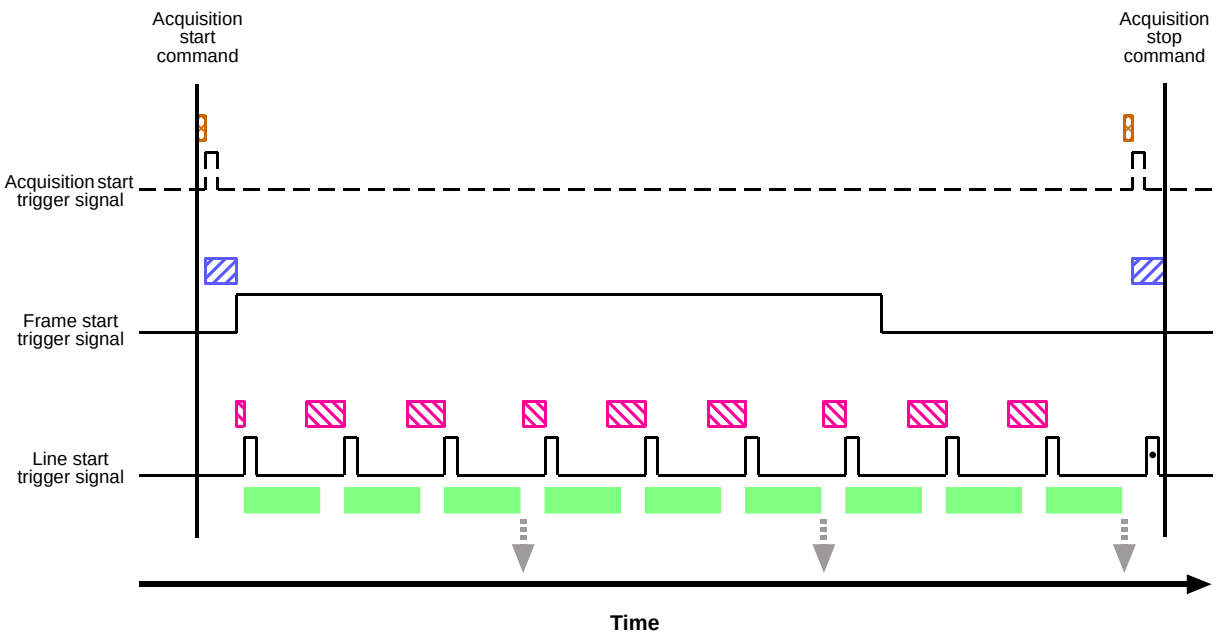



Fig. 30: Use Case Diagram - Continuous Frame Mode with Acquisition Start Triggering Set to Off, Frame Start and Line Start Triggering Set to On, and Partial Closing Frame set to False

## Use Case 6 - Acquisition Start Triggering Off (Free Run), Frame Start and Line Start Triggering On, Frame Start Trigger Level High, Partial Closing Frame True

Use case six is illustrated on page 96.

This use case is equivalent to the preceding use case five, except for the fact that Partial Closing Frame is set to True.

In this use case, the Acquisition Start Trigger Mode parameter is set to off. The camera will internally manage acquisition start trigger signals without any need for triggering by the user ("free run").

The Frame Start Trigger Mode and the Line Start Trigger Mode parameters are set to on, requiring that frame start and line start trigger signals are applied to the camera.

The Frame Start Trigger Activation is set to Level High. This means that a transition of the frame start trigger signal is always present as long as the signal stays high. Accordingly, during this period frames can be acquired without interruption which otherwise will happen if a related preceding transition of the frame start trigger signal has not occurred (c.f. also use case four).

In this example, each frame is set to include three lines.

In this example, the frame start trigger signal goes low while a frame is being acquired (i.e. while line one of the closing frame of the sequence of frames is being acquired). With Partial Closing Frame set to true, only the partial closing frame will be acquired and transmitted. In this example, the partial closing frame includes only one line.

When the Acquisition Mode set to Continuous Frame, the camera will be set to acquire frames until an acquisition stop command is issued.

**Settings:**
Acquisition Mode = Continuous Frame

Acquisition Start Trigger Mode = Off

Frame Start Trigger Mode = On
Frame Start Trigger Source = Line 2
Frame Start Trigger Activation = Level High
Partial Closing Frame = True
Lines Per Frame (Height) = 3

Line Start Trigger Mode = On
Line Start Trigger Source = Line 3
Line Start Trigger Activation = Rising Edge

= trigger signal internally generated by the camera

= trigger signal applied by the user

= camera is waiting for an acquisition start trigger signal

= camera is waiting for a frame start trigger signal

= camera is waiting for a line start trigger signal

= line exposure and readout

= line start trigger signal is ignored because the camera is waiting for a frame start trigger signal

= complete frame transmitted

= partial frame transmitted

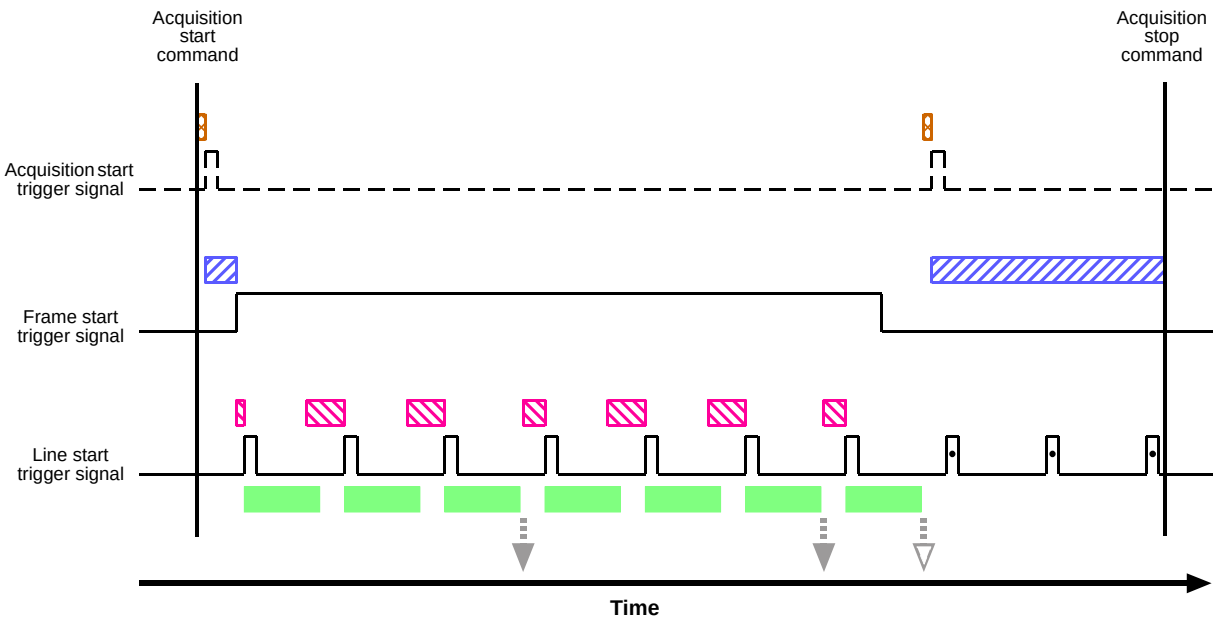



Fig. 31: Use Case Diagram - Continuous Frame Mode with Acquisition Start Triggering Set to Off, Frame Start and Line Start Triggering Set to On, and Partial Closing Frame set to True

## Use Case 7 - Acquisition Start and Frame Start Triggering Off (Free Run), Line Start Triggering On

Use case seven is illustrated on page 98.

This use case is equivalent to use case two, except for the fact that the Line Start Trigger Mode parameter is set to on.

In this use case, the Acquisition Start Trigger Mode and the Frame Start Trigger Mode parameters are set to off. The camera will internally manage acquisition start and frame start trigger signals without any need for triggering by the user ("free run").

The Line Start Trigger Mode parameter is set to on, requiring that a line start trigger signal is applied to the camera.

The rate at which the camera will acquire lines will be determined by the line start trigger signal and must be below the maximum allowed line rate determined by the current setting.

In this example, each frame is set to include three lines.

When the Acquisition Mode is set to Continuous Frame, the camera will be set to acquire frames until an acquisition stop command is issued.

If an Acquisition Stop command is issued when not all lines of the current frame are yet acquired, the partial frame will be transmitted.

**Settings:**
Acquisition Mode = Continuous Frame

Acquisition Start Trigger Mode = Off

Frame Start Trigger Mode = Off
Lines Per Frame (Height) = 3

Line Start Trigger Mode = On
Line Start Trigger Source = Line 3
Line Start Trigger Activation = Rising Edge

= trigger signal internally generated by the camera

= trigger signal applied by the user

= camera is waiting for an acquisition start trigger signal

= camera is waiting for a frame start trigger signal

= camera is waiting for a line start trigger signal

= line exposure and readout

= complete frame transmitted

= partial frame transmitted

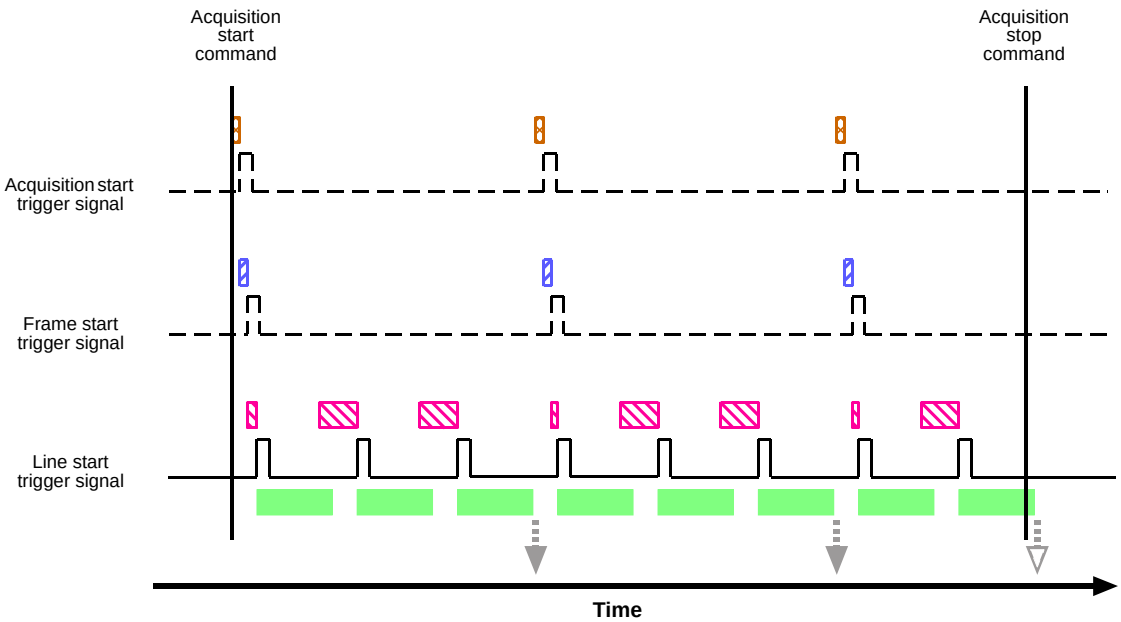



Fig. 32: Use Case Diagram - Continuous Frame Mode with Acquisition Start and Frame Start Triggering Set to Off and Line Start Triggering Set to On

## Use Case 8 - Acquisition Start Triggering On, Frame Start and Line Start Triggering Off (Free Run)

Use case eight is illustrated on page 100.

In this use case, the Acquisition Start Trigger Mode parameter is set to on, requiring that an acquisition start trigger signal is applied to the camera.

The Frame Start Trigger Mode and the Line Start Trigger Mode parameters are set to off. The camera will internally manage frame start and line start trigger signals without any need for triggering by the user ("free run").

In this example, Acquisition Frame Count is set to two. Accordingly, two consecutive frames will be acquired for each transition of the acquisition start trigger signal.

The rate at which the camera will acquire lines will normally be determined by the camera's Acquisition Line Rate Abs parameter. If the Acquisition Line Rate Abs parameter is disabled, the camera will acquire lines at the maximum allowed line rate. Note that the overall line rate will also depend on the acquisition start trigger signal: Lines will only be acquired after a related preceding transition of the acquisition start trigger signal has occurred.

In this example, each frame is set to include three lines.

When the Acquisition Mode is set to Continuous Frame, the camera will be set to acquire frames until an acquisition stop command is issued.

**Settings:**
Acquisition Mode = Continuous Frame

Acquisition Start Trigger Mode = On
Acquisition Start Trigger Source = 1
Acquisition Start Trigger Activation = Rising Edge
Acquisition Frame Count = 2

Frame Start Trigger Mode = Off
Lines Per Frame (Height) = 3

Line Start Trigger Mode = Off

= trigger signal internally generated by the camera

= trigger signal applied by the user

= camera is waiting for an acquisition start trigger signal

= camera is waiting for a frame start trigger signal

= camera is waiting for a line start trigger signal

= line exposure and readout

= frame transmitted

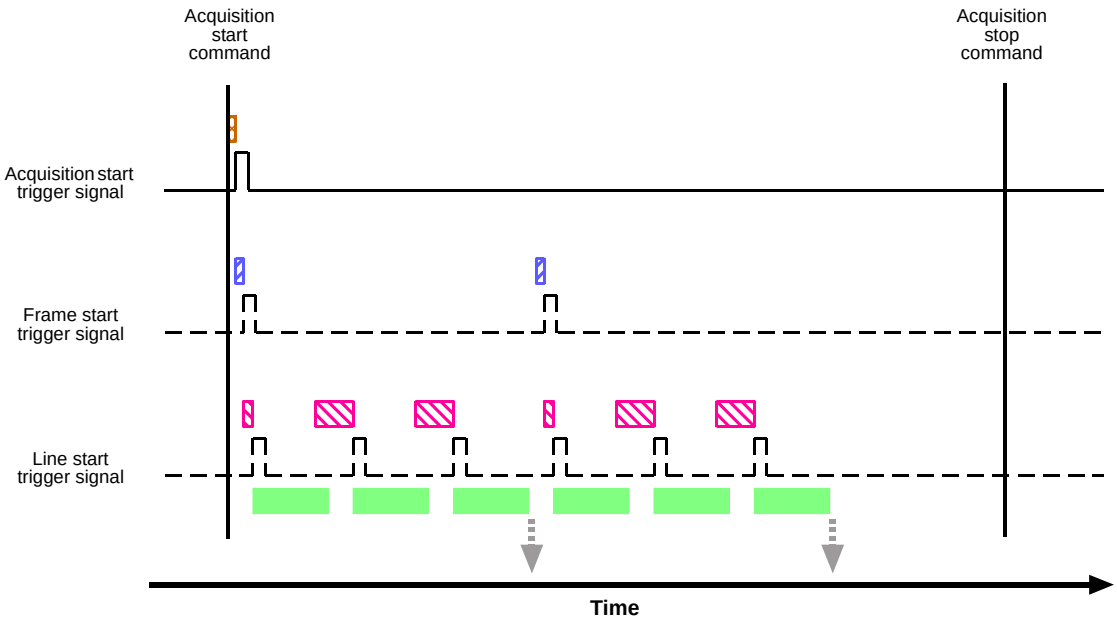



Fig. 33: Use Case Diagram - Continuous Frame Mode with Acquisition Start Triggering Set to On and Frame Start and Line Start Triggering Set to Off

### Use Case 9 - Acquisition Start and Line Start Triggering On, Frame Start Triggering Off (Free Run)

Use case nine is illustrated on page 102.

In this use case, the Acquisition Start Trigger Mode and the Line Start Trigger Mode parameters are set to on, requiring that an acquisition start and a line start trigger signal are applied to the camera.

The Frame Start Trigger Mode parameter is set to off. The camera will internally manage frame start signals without any need for triggering by the user ("free run").

In this example, Acquisition Frame Count is set to two. Accordingly, two consecutive frames will be acquired for each transition of the acquisition start trigger signal.

The rate at which the camera will acquire lines will be determined by the line start trigger signal and must be below the maximum allowed line rate determined by the current setting. Note that the overall line rate will also depend on the acquisition start trigger signal: Lines will only be acquired after a related preceding transition of the acquisition start trigger signal has occurred.

In this example, each frame is set to include three lines.

When the Acquisition Mode is set to Continuous Frame, the camera will be set to acquire frames until an acquisition stop command is issued.

**Settings:**
Acquisition Mode = Continuous Frame

Acquisition Start Trigger Mode = On
Acquisition Start Trigger Source = 1
Acquisition Start Trigger Activation = Rising Edge
Acquisition Frame Count = 2

Frame Start Trigger Mode = Off
Lines Per Frame (Height) = 3

Line Start Trigger Mode = On
Line Start Trigger Source = Line 3
Line Start Trigger Activation = Rising Edge

= trigger signal internally generated by the camera

= trigger signal applied by the user

= camera is waiting for an acquisition start trigger signal

= camera is waiting for a frame start trigger signal

= camera is waiting for a line start trigger signal

= line exposure and readout

= line start trigger signal is ignored because the camera is waiting for an acquisition start trigger signal

= complete frame transmitted

= partial frame transmitted

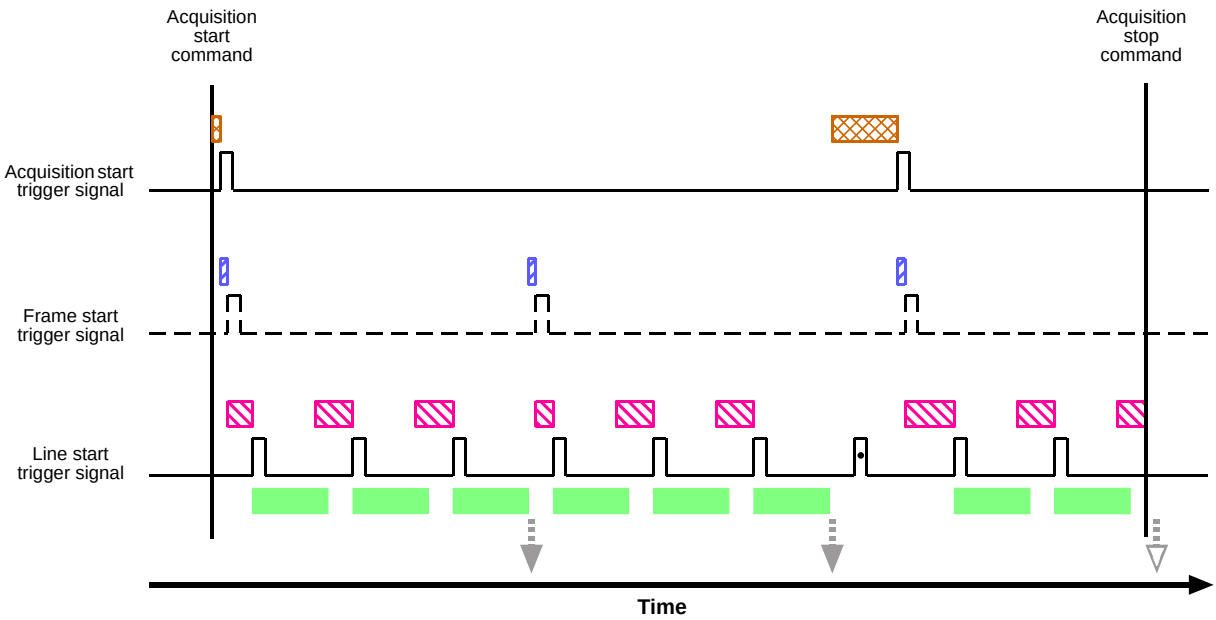



Fig. 34: Use Case Diagram - Continuous Frame Mode with Acquisition Start and Line Start Triggering Set to On and Frame Start Triggering Set to Off

# 8.3 The Shaft Encoder Module

The camera is equipped with a shaft encoder software module. The module can accept input from a two channel shaft encoder (Phase A and Phase B). The module outputs a signal that can be used, for example, as a source signal for the line start trigger function or the frame start trigger function in the camera. Fig. 35 shows a typical implementation of the shaft encoder software module in the camera.

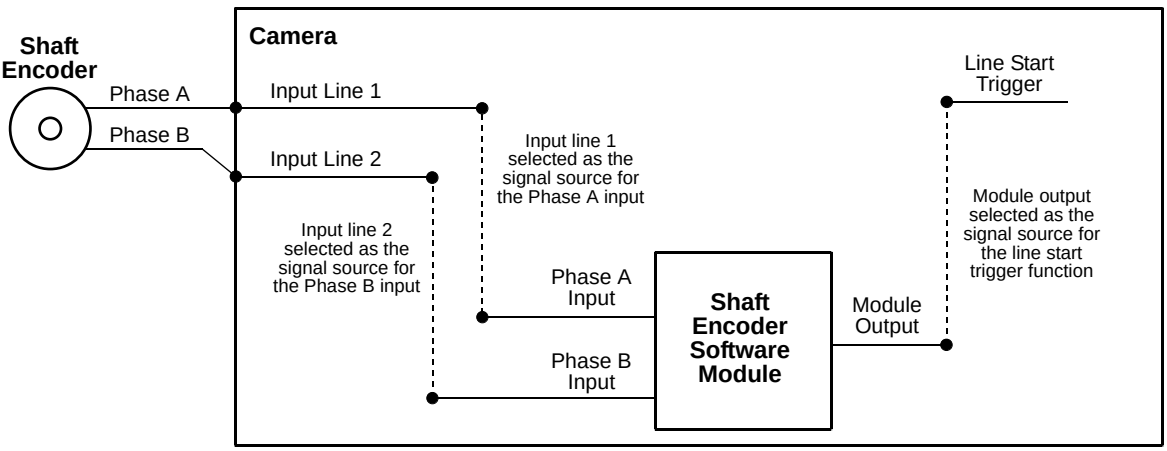

Shaft Encoder
Phase A
Phase B
Camera
Input Line 1
Input Line 2
Input line 1 selected as the signal source for the Phase A input
Input line 2 selected as the signal source for the Phase B input
Phase A Input
Phase B Input
Shaft Encoder Software Module
Module Output
Line Start Trigger
Module output selected as the signal source for the line start trigger function

Fig. 35: Typical Shaft Encoder Module Implementation

To use the shaft encoder module, you must select a source signal for the Phase A input and for the Phase B input on the module. The allowed source signals for the Phase A and Phase B module inputs are camera input line 1, camera input line 2, and camera input line 3. So, for example, you could apply the Phase A signal from a shaft encoder to physical input line 1 of the camera and select input line 1 as the source signal for the Phase A input to the module. And you could apply the Phase B signal from a shaft encoder to physical input line 2 of the camera and select input line 2 as the source signal for the Phase B input to the module. More information about selecting a source signal for the module inputs appears in a code snippet later in this section.

Fig. 36 shows how the software module will interpret the input from the shaft encoder when the encoder is connected as illustrated in Fig. 35. The software module will sense forward ticks from the encoder when the input is as shown in the left part of Fig. 36. The software module will sense reverse ticks from the encoder when the input is as shown in the right part of the Fig. 36.

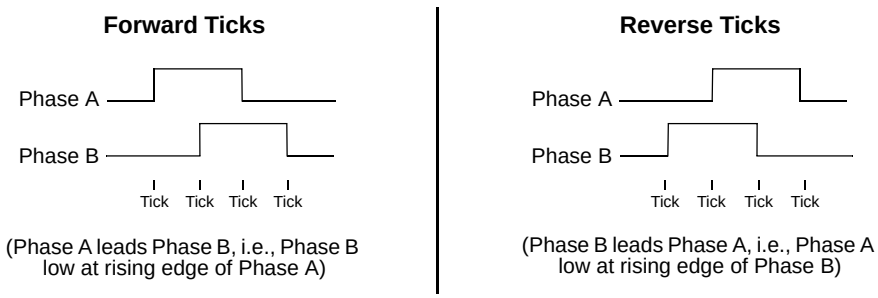

Forward Ticks
Phase A
Phase B
Tick Tick Tick Tick
(Phase A leads Phase B, i.e., Phase B low at rising edge of Phase A)
Reverse Ticks
Phase A
Phase B
Tick Tick Tick Tick
(Phase B leads Phase A, i.e., Phase A low at rising edge of Phase B)

Fig. 36: Software Module Direction Sensing

Note that if this interpretation of direction is not as you desire, you could change it by moving the Phase A output from the shaft encoder to input line 2 and the Phase B output to input line 1.

## Shaft Encoder Module Parameters

There are several parameters and commands associated with the shaft encoder module. The list below describes the parameters and commands and explains how they influence the operation of the module.

- The **Shaft Encoder Module Counter Mode** parameter controls the tick counter on the shaft encoder module. The tick counter counts the number of ticks that have been received by the module from the shaft encoder. This parameter has two possible values: Follow Direction and Ignore Direction.

  If the mode is set to Follow Direction, the counter will increment when the module receives forward ticks from the shaft encoder and will decrement when it receives reverse ticks.

  If the mode is set to Ignore Direction, the counter will increment when it receives either forward ticks or reverse ticks.
- The **Shaft Encoder Module Counter** parameter indicates the current value of the tick counter. This is a read only parameter.
- The **Shaft Encoder Counter Module Max** parameter sets the maximum value for the tick counter. The minimum value for this parameter is 0 and the maximum is 32767.

  If the counter is incrementing and it reaches the max, it will roll over to 0. That is:
  Max + 1 = 0

  If the counter is decrementing and it reaches 0, it will roll back to the max. That is:
  0 - 1 = Max
- The **Shaft Encoder Module Counter Reset** command resets the tick counter count to 0.
- The **Shaft Encoder Module Mode** parameter controls the behavior of the "reverse counter" that is built into the module. This parameter has two possible values: Any Direction and Forward Only. For more information about this parameter, see the detailed description of the reverse counter that appears later in this section.
- The **Shaft Encoder Module Reverse Counter Max** parameter sets a maximum value for the module's "reverse counter". The minimum value for this parameter is 0 and the maximum is 32767. For more information about this parameter, see the detailed description of the reverse counter that appears later in this section.
- The **Shaft Encoder Module Reverse Counter Reset** command resets the reverse counter count to 0 and informs the software module that the current direction of conveyor movement is forward. For more information about this parameter, see the detailed description of the reverse counter that appears later in this section.

## Setting the Shaft Encoder Module Parameters

To use the shaft encoder software module effectively, you should do the following:

- Select a signal source for the Phase A and Phase B inputs on the module.

  (By default, input line 1 is selected as the signal source for the Phase A input and input line 2 is selected as the signal source for the Phase B input.)
- Make sure that the output from the encoder module is selected as the signal source for a camera function. Currently, output from the encoder module can be selected as the signal source for the camera's Frame Start Trigger function or for the camera's Line Start Trigger function.
- Set the Shaft Encoder Module Counter Mode and the Shaft Encoder Module Mode as appropriate.

You can set the encoder module parameter values, issue commands to the encoder module, and select signal sources from within your application software by using the pylon API. The code snippet below illustrates using the API to set the parameter values and to issue commands to the encoder module.

```
// Select physical input line 1 as the source signal for the Phase A input on the module
// and physical input line 2 as the source signal for the Phase B input
Camera.ShaftEncoderModuleLineSelector.SetValue( ShaftEncoderModuleLineSelector_PhaseA );
Camera.ShaftEncoderModuleLineSource.SetValue( ShaftEncoderModuleLineSource_Line1 );
Camera.ShaftEncoderModuleLineSelector.SetValue( ShaftEncoderModuleLineSelector_PhaseB );
Camera.ShaftEncoderModuleLineSource.SetValue( ShaftEncoderModuleLineSource_Line2 );

// Enable the camera's Line Start Trigger function and select the outout from the encoder
// module as the source signal for the Line Start Trigger
Camera.TriggerSelector.SetValue( TriggerSelector_LineStart );
Camera.TriggerMode.SetValue( TriggerMode_On );
Camera.TriggerSource.SetValue ( TriggerSource_ShaftEncoderModuleOut );
Camera.TriggerActivation.SetValue( TriggerActivation_RisingEdge );

// Set the shaft encoder module counter mode
Camera.ShaftEncoderModuleCounterMode.SetValue( ShaftEncoderModuleCounterMode_FollowDirection );

// Set the shaft encoder module mode
Camera.ShaftEncoderModuleMode.SetValue( ShaftEncoderModuleMode_AnyDirection );

// Set the shaft encoder module counter max and the shaft encoder module reverse counter max
Camera.ShaftEncoderModuleCounterMax.SetValue( 32767 );
Camera.ShaftEncoderModuleReverseCounterMax.SetValue( 15 );

// Get the current value of the shaft encoder module counter
int64_t encodercounterSize = Camera.ShaftEncoderModuleCounter.GetValue();

// Reset the shaft encoder module counter and the shaft encoder module reverse counter
Camera.ShaftEncoderModuleCounterReset.Execute( );
Camera.ShaftEncoderModuleReverseCounterReset.Execute( );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

For more information about the line start trigger, see Section 8.2.4 on page 77.

## The Reverse Counter

The main purpose of the reverse counter is to compensate for mechanical "jitter" in the conveyor used to move objects past the camera. This jitter usually manifests itself as a momentary change in the direction of the conveyor.

The rules that govern the operation of the reverse counter are as follows:

- If the conveyor is running in the reverse direction and the current reverse counter count is less than the maximum (i.e., less than the current setting of the Reverse Counter Max parameter), the reverse counter will increment once for each shaft encoder reverse tick received.
- If the conveyor is running in the forward direction and the current reverse counter count is greater than zero, the reverse counter will decrement once for each shaft encoder forward tick received
- When the Shaft Encoder Mode is set to Forward Only:

  If the reverse counter is not incrementing or decrementing, the software module will output a trigger signal for each forward tick received from the shaft encoder.

  If the reverse counter is incrementing or decrementing, trigger signal output will be suppressed.
- When the Shaft Encoder Mode is set to Any Direction:

  If the reverse counter is not incrementing or decrementing, the software module will output a trigger signal for each forward tick or reverse tick received from the shaft encoder.

  If the reverse counter is incrementing or decrementing, trigger signal output will be suppressed.

To understand how these rules affect the operation of the encoder software module, consider the following cases:

### Case 1

This is the simplest case, i.e., the Shaft Encoder Reverse Counter Max is set to zero. In this situation, the reverse counter never increments or decrements and it will have no effect on the operation of the encoder software module.

When the Shaft Encoder Reverse Counter Max is set to zero:

- If the Shaft Encoder Module Mode is set to Forward Only, the software module will output a trigger signal whenever it receives a forward tick from the shaft encoder, but not when it receives a reverse tick.
- If the Shaft Encoder Module Mode is set to Any Direction, the software module will output a trigger signal whenever it receives either a forward tick or a reverse tick from the shaft encoder.

**Case 2**

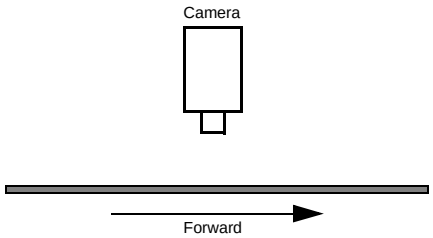
In this case, assume that:

- A shaft encoder is attached to a conveyor belt that normally moves continuously in the forward direction past a camera.
- The conveyor occasionally "jitters" and when it jitters, it moves in reverse for 4 or 5 ticks.

For this case, the Shaft Encoder Module Mode parameter should be set to Forward Only. The Shaft Encoder Module Reverse Counter Max should be set to a value that is higher than the jitter we expect to see. We decide to set the value to 10.

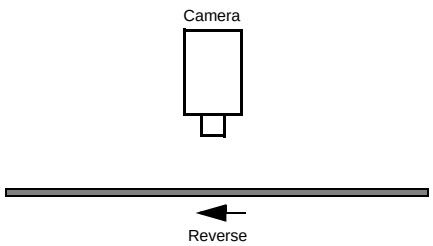
Given this situation and these settings, the series of diagrams below explains how the encoder software module will act:

1



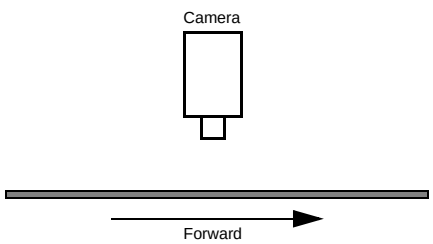

The conveyor is moving forward and the encoder is generating forward ticks. Whenever the module receives a forward tick, it outputs a trigger signal.
The reverse counter is at 0.

2




The conveyor jitters and moves briefly in reverse. During this reverse movement, the shaft encoder generates 5 reverse ticks.
The reverse counter will increment by 1 for each reverse tick and when the reverse motion stops, the reverse counter count will be 5.
While the reverse counter is incrementing, the output of trigger signals from the module is suppressed.

3




The conveyor resumes forward motion and the shaft encoder module begins generating forward ticks.
The reverse counter will decrement by 1 for each forward tick. While the reverse counter is decrementing, the output of trigger signals from the module is suppressed.
When the reverse counter decrements to 0, decrementing stops and suppression of the trigger signals stops. The module will begin outputting a trigger signal for each forward tick received.

By suppressing trigger signals when the conveyor was moving in reverse and then suppressing an equal number of trigger signals when forward motion is resumed, we ensure that the conveyor is in its "pre-jitter" position when the module begins generating trigger signals again.

Note in step two that if the conveyor runs in reverse for a long period and the reverse counter reaches the max setting, the counter simply stops incrementing. If the conveyor continues in reverse, no output triggers will be generated because the Shaft Encoder Mode is set to Forward only.

### Case 3

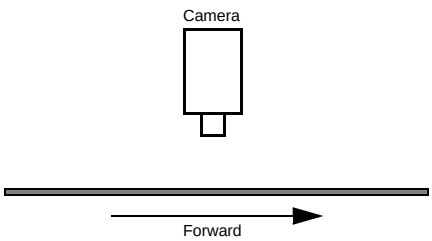
In this case, assume that:

- We are working with a small conveyor that moves back and forth in front of a camera.
- A shaft encoder is attached to the conveyor.
- The conveyor moves in the forward direction past the camera through its complete range of motion, stops, and then begins moving in reverse.
- The conveyor moves in the reverse direction past the camera through its complete range of motion, stops, and then begins moving forward.
- This back an forth motion repeats.
- The conveyor occasionally "jitters". When it jitters, it moves 4 or 5 ticks in a direction of travel opposite to the current normal direction.

For this case, the Shaft Encoder Module Mode parameter should be set to Any Direction. The Shaft Encoder Module Reverse Counter Max should be set to a value that is higher than the jitter we expect to see. We decide to set the value to 10.

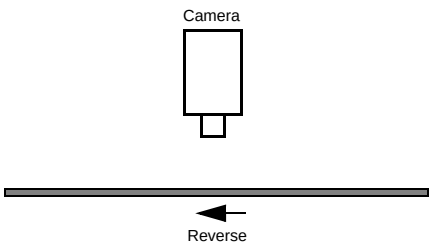
Given this situation and these settings, this series of diagrams explains how the encoder software module will act during conveyor travel:

1



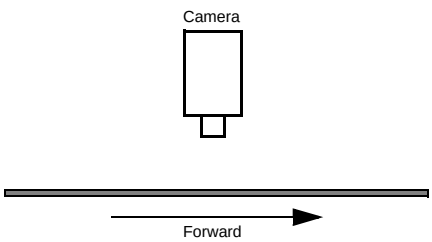

The conveyor is moving forward and the encoder is generating forward ticks. Whenever the module receives a forward tick, it outputs a trigger signal.
The reverse counter is at 0.

2



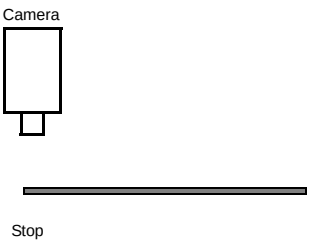

The conveyor jitters and moves briefly in reverse. During this reverse movement, the shaft encoder generates 5 reverse ticks.
The reverse counter will increment by 1 for each reverse tick and when the reverse motion stops, the reverse counter count will be 5.
While the reverse counter is incrementing, the output of trigger signals from the module is suppressed.

3



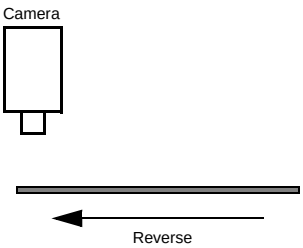

The conveyor resumes forward motion and the shaft encoder module begins generating forward ticks.
The reverse counter will decrement by 1 for each forward tick. While the reverse counter is decrementing, the output of trigger signals from the module is suppressed
When the reverse counter decrements to 0, decrementing stops and suppression of the trigger signals stops. The module will begin outputting a trigger signal for each forward tick received.

4




The conveyor reaches the end of its forward travel and it stops.

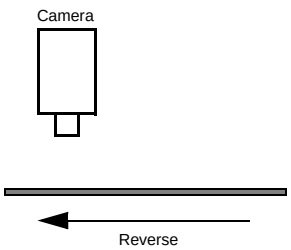
5




The conveyor begins moving in reverse and the shaft encoder starts generating reverse ticks.

The reverse counter will increment by 1 for each reverse tick. While the reverse counter is incrementing and the reverse count is below the max (10 in this case), the output of trigger signals from the module is suppressed

6

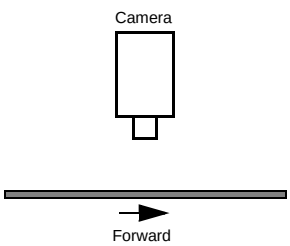



The reverse counter reaches the max (10 in this case) and stops incrementing.

Suppression of trigger signals is ended. Because the shaft encoder mode is set to any direction, the module begins generating one trigger signal for each reverse tick received.

The reverse counter remains at 10.

7

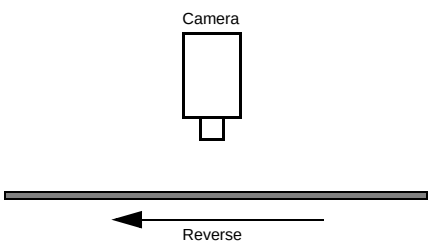



The conveyor jitters and moves forward briefly. During this forward movement, the shaft encoder generates 4 forward ticks.

The reverser counter will decrement by 1 for each forward tick. When the forward motion stops, the reverse counter count will be 6.

While the reverse counter is decrementing, the output of trigger signals from the module is suppressed.

8

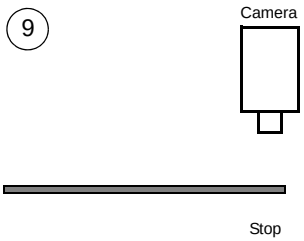



The conveyor resumes reverse motion and the shaft encoder module begins generating reverse ticks.

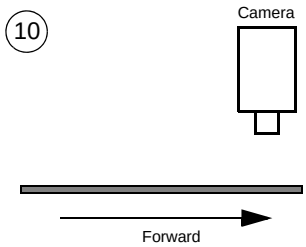
The reverse counter will increment by 1 for each reverse tick. While the reverse counter is incrementing, the output of trigger signals from the module is suppressed.

When the reverse counter reaches the max (10 in this case) it stops incrementing and suppression of the trigger signals stops. The module will resume outputting a trigger signal for each reverse tick received.

The reverse counter count is now 10.

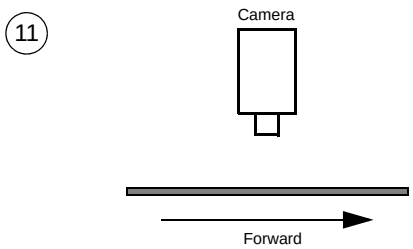



The conveyor reaches the end of its reverse travel and it stops.




The conveyor begins moving forward and the shaft encoder starts generating forward ticks.

The reverse counter is at 10 and will now begin decrementing by 1 for each forward tick. While the reverse counter is decrementing and the reverse count is greater than 0, the output of trigger signals from the module is suppressed




The reverse counter reaches 0.

Suppression of trigger signals is ended. Because the shaft encoder mode is set to any direction, the module begins generating one trigger signal for each forward tick received.

The reverse counter remains at 0.

There are two main things to notice about this example. First, because the encoder mode is set to any direction, ticks from the shaft encoder will cause the module to output trigger signals regardless of the conveyor direction, as long as the reverse counter is not incrementing or decrementing. Second, the reverse counter will compensate for conveyor jitter regardless of the conveyor direction.

It is important to reset the reverse counter before the first traverse in the forward direction. A reset sets the counter to 0 and synchronizes the counter software with the conveyor direction. (The software assumes that the conveyor will move in the forward direction after a counter reset).

# 8.4 Frequency Converter

The camera is equipped with a frequency converter module that allows triggering the camera at a frequency that differs from the frequency of the input signals received.

The module can accept input signals from one of the three input lines or signals (ticks) from the shaft encoder module.

The frequency converter module includes three sub-modules acting in sequence on the original signals:

- The pre-divider module receives the input signals. The module allows employing an integer factor, the pre-divider, to decrease the original frequencies and passes the signals on to the next module, the multiplier module.

  If for example a pre-divider of 2 is selected only every other input signal is passed out unchanged to the multiplier module and, accordingly, the frequency is halved. If a pre-divider of 1 is selected every input signal is passed out unchanged to the multiplier module.

  Employing the pre-divider may be advisable for decreasing periodic jitter of the input signals and will be required if the input signal frequency is higher than 100 kHz. The signal frequency of the signals passed on to the multiplier module must be within the range of 10 Hz to 100 kHz. Periodic jitter is likely to be present when input signals from the shaft encoder are accepted.

  We recommend to only use low values for the pre-divider. The original signal frequency should be changed as little as possible to facilitate frequency adjustment by the multiplier module.

- The multiplier module receives the signals from the pre-divider module. The signal frequency must be within the range of 10 Hz to 100 kHz. The multiplier module allows applying an integer factor, the multiplier, to generate signals at increased frequencies and passes the signals on to the next module, the post-divider module.

  If, for example, a multiplier of 2 is selected signals are generated at double the frequency of the signals received from the pre-divider module and are passed on to the divider module. If a multiplier of 1 is selected every signal received from the pre-divider module is passed unchanged on to the divider module.

  The Align parameter can be set to "rising edge" and "falling edge". If "rising edge" is selected there will be for the rising edge of each signal received from the pre-divider module a phase-locked, matching rising edge among the signals generated. If "falling edge" is selected there will be for the falling edge of each signal received from the pre-divider module a phase-locked, matching falling edge among the signals generated.

  Make sure to select a multiplier that will not too much increase the frequency such that the camera will be overtriggered. Temporarily, a too high frequency may occur during frequency adjustment causing overtriggering even if a relatively low multiplier was selected. The PreventOvertrigger parameter provides a safeguard against overtriggering the camera. We recommend setting the PreventOvertrigger parameter to True to prevent overtriggering.

- The post-divider module receives the signals from the multiplier module. The post-divider module allows employing an integer factor, the post-divider, to generate signals at decreased frequencies and provides these signals to be used as camera trigger signals, e.g. as line start triggers.

  If for example a post-divider of 2 is selected only every other signal received from the multiplier module is passed out from the divider module and, accordingly, the frequency is halved. If a post-divider of 1 is selected every signal received from the multiplier module is passed out un-

changed from the divider module.

| | |
|---|---|
| | You can use the frequency converter to multiply the original signal frequency by a fractional value. We recommend multiplying the frequency by the enumerator value using the multiplier module and dividing the resulting frequency by the denominator value using the post-divider module. |

You can configure the frequency converter module from within your application by using a dynamic API. The following code snippet illustrates setting parameter values:

```
INodeMap &Control = *Camera.GetNodeMap();

// possible values for FrequencyConverterInputSource:
// Line1
// Line2
// Line3
// ShaftEncoderModuleOut
CEnumerationPtr(Control.GetNode("FrequencyConverterInputSource"))-
>FromString("ShaftEncoderModuleOut");

// ranges for divider and multiplier:
// divider   : 1...128
// multiplier: 1...32
CIntegerPtr(Control.GetNode("FrequencyConverterPreDivider"))->SetValue(4);
CIntegerPtr(Control.GetNode("FrequencyConverterMultiplier"))->SetValue(17);
CIntegerPtr(Control.GetNode("FrequencyConverterPostDivider"))->SetValue(1);
```

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the shaft encoder module see, Section  on page 103.

# 8.5 Acquisition Monitoring Tools

The camera includes the acquisition status feature and generates four output signals that you can use to monitor the progress of line and frame acquisition by the camera: the exposure active signal, the acquisition trigger wait signal, the frame trigger wait signal, and the line trigger wait signal.

The camera also allows selecting the output of the frequency converter module or the shaft encoder module as output signals.

## 8.5.1 Exposure Active Signal

The camera's Exposure Active output signal will go high when the exposure time for each line acquisition begins and goes low when the exposure time ends. An example of the Exposure Active signal's behavior on a camera using a rising edge external line start trigger signal (ExLSTrig) and the timed exposure mode is shown in Fig. 37.

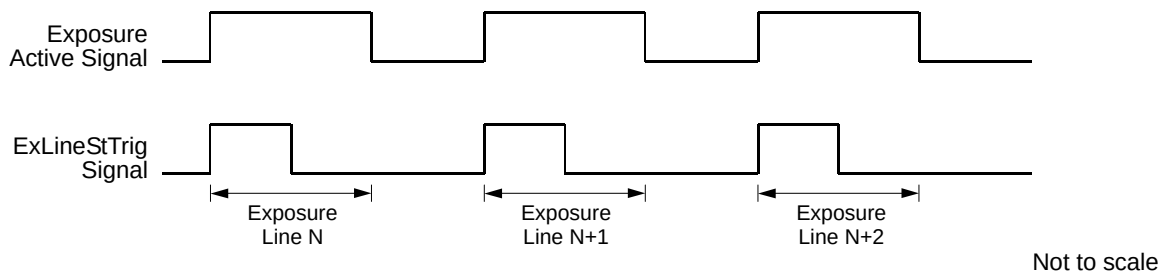



Fig. 37: Exposure Active Signal

By default, the Exposure Active signal is selected as the source signal for output line 1 on the camera. However, the selection of the source signal for a physical output line can be changed.

For more information about selecting the source signal for an output line on the camera, see Section 7.7.2.3 on page 65.

For more information about the electrical characteristics of the camera's output lines, see Section 7.7.2 on page 62.

## 8.5.2 Acquisition Status

If a camera receives a software acquisition start trigger signal when it is not in a "waiting for acquisition start trigger" acquisition status, it will simply ignore the trigger signal and will generate an acquisition start overtrigger event.

If a camera receives a software frame start trigger signal when it is not in a "waiting for frame start trigger" acquisition status, it will simply ignore the trigger signal and will generate a frame start overtrigger event.

If a camera receives a software line start trigger signal when it is not in a "waiting for line start trigger" acquisition status, it will simply ignore the trigger signal and will generate a line start overtrigger event.

The camera's acquisition status indicator gives you the ability to check whether the camera is in a "waiting for acquisition start trigger" acquisition status or in a "waiting for frame start trigger" acquisition status or in a "waiting for line start trigger" acquisition status. If you check the acquisition status before you apply each software acquisition start trigger signal, software frame start trigger signal and software line start trigger signal, you can avoid applying trigger signals to the camera that will be ignored.

The acquisition status indicator is designed for use when you are using host control of image acquisition, i.e., when you are using software acquisition start, frame start, and line start trigger signals.

To determine the acquisition status of the camera via the Basler pylon API:

- Use the Acquisition Status Selector to select the Acquisition Trigger Wait status or the Frame Trigger Wait status or the Line Trigger Wait status.
- Read the value of the Acquisition Status parameter.

  If the value is set to "false", the camera is not waiting for the trigger signal.

  If the value is set to "true", the camera is waiting for the trigger signal.

You can check the acquisition status from within your application software by using the Basler pylon API. The following code snippet illustrates using the API to check the acquisition status:

```
// Check the acquisition start trigger acquisition status
// Set the acquisition status selector
Camera.AcquisitionStatusSelector.SetValue
( AcquisitionStatusSelector_AcquisitionTriggerWait );
// Read the acquisition status
bool IsWaitingForAcquisitionTrigger = Camera.AcquisitionStatus.GetValue();

// Check the frame start trigger acquisition status
// Set the acquisition status selector
Camera.AcquisitionStatusSelector.SetValue
( AcquisitionStatusSelector_FrameTriggerWait );
// Read the acquisition status
bool IsWaitingForFrameTrigger = Camera.AcquisitionStatus.GetValue();
```

```
// Check the line start trigger acquisition status
// Set the acquisition status selector
Camera.AcquisitionStatusSelector.SetValue
( AcquisitionStatusSelector_LineTriggerWait );
// Read the acquisition status
bool IsWaitingForLineTrigger = Camera.AcquisitionStatus.GetValue();
```

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon API and the pylon Viewer, see Section 3 on page 17.

### 8.5.3 Acquisition Trigger Wait Signal

The camera's Acquisition Trigger Wait output signal will be low when the camera is in the process of acquiring a frame and is not able to accept a new acquisition start trigger. As soon as the camera is ready to accept a new acquisition start trigger, the Acquisition Trigger Wait signal will go high.

This signal can be selected as the source signal for one of the output lines on the camera.

For more information about selecting the source signal for an output line on the camera, see Section 7.7.2.3 on page 65.

For more information about the electrical characteristics of the camera's output lines, see Section 7.7.2 on page 62.

### 8.5.4 Frame Trigger Wait Signal

The camera's Frame Trigger Wait output signal will be low when the camera is in the process of acquiring a frame and is not able to accept a new frame start trigger. As soon as the current frame acquisition is complete and the camera is ready to acquire a new frame, the signal will go high.

By default, the Frame Trigger Wait signal is selected as the source signal for output line 2 on the camera. However, the selection of the source signal for a physical output line can be changed.

For more information about selecting the source signal for an output line on the camera, see Section 7.7.2.3 on page 65.

For more information about the electrical characteristics of the camera's output lines, see Section 7.7.2 on page 62.

### 8.5.5 Line Trigger Wait Signal

The camera's Line Trigger Wait output signal will be low when the camera is in the process of acquiring a line and is not able to accept a new line start trigger. As soon as the current line acquisition is complete and the camera is ready to acquire a new line, the Line Trigger Wait signal will go high.

This signal can be selected as the source signal for one of the output lines on the camera.

For more information about selecting the source signal for an output line on the camera, see Section 7.7.2.3 on page 65.

For more information about the electrical characteristics of the camera's output lines, see Section 7.7.2 on page 62.

### 8.5.6 Input Related Signals as Output Signals

The camera allows selecting the output signals of the shaft encoder module or of the frequency converter module and assigning them to one of the camera's digital output lines. In this fashion input signals can be passed through a camera to trigger additional cameras.

In this case, setting a minimum output pulse width may be necessary to ensure output signal detection.

For more information about selecting the source signal for an output line on the camera, see Section 7.7.2.3 on page 65.

For more information about the electrical characteristics of the camera's output lines, see Section 7.7.2 on page 62.

For more information about the minimum output pulse width feature, see Section 7.6.2.2 on page 58.

# 8.6 Frame Transmission Time

As mentioned in earlier sections of this chapter, each time that a complete frame has been accumulated in the camera's frame memory, the frame will be transmitted from the camera to your host PC via the camera's Ethernet network connection. The image data in the frame will be packetized and transmitted in compliance with the mechanisms described in the GigE Vision standard.

For more detailed information about receiving the frames as they arrive in your host PC, refer to the Basler pylon Programmer's Guide and API Reference. The sample programs included with the pylon software development kit (SDK) also provide more detailed information about handling incoming image data in your host PC.

For more information about managing the bandwidth of the Ethernet network connection between your camera(s) and your host PC, see Section 5 on page 31.

You can calculate the approximate time that it will take to transmit a frame from the camera to the host PC by using this formula:

$$\sim \text{Frame Transmission Time} = \frac{\text{Payload Size Parameter Value}}{\text{Device Current Throughput Parameter Value}}$$

Note that this is an approximate frame transmission time. Due to the nature of the Ethernet network, the transmission time could vary.

Due to the nature of the Ethernet network, there can be a delay between the point where a complete frame is acquired and the point where transmission of the acquired frame begins. This start delay can vary from frame to frame. The start delay, however, is of very low significance when compared to the transmission time.

For more information about the Payload Size and Device Current Throughput parameters, see Section 5.1 on page 31.

# 8.7 Maximum Allowed Line Acquisition Rate

In general, the maximum allowed line acquisition rate can be limited by three factors:

- The amount of time it takes to read an acquired line out of the imaging sensor and into the camera's frame buffer. Since readout time is fixed, it establishes an absolute maximum for the line rate. Note that the readout time stays the same regardless of the Width parameter setting for the frame.
- The exposure time for acquired lines. If you use longer exposure times, you can acquire fewer lines per second.
- The amount of time that it takes to transmit a completed frame from the camera to your host PC. The amount of time needed to transmit a frame depends on the bandwidth assigned to the camera.

When the camera's acquisition mode is set to single frame, the maximum possible acquisition frame rate can not be achieved. This is true because the camera performs a complete internal setup cycle for each single frame.

To determine the maximum allowed line acquisition rate with your current camera settings, you can use a parameter called the Resulting Line Rate. The Resulting Line Rate parameter indicates the camera's current maximum allowed line acquisition rate taking the readout time, exposure time, and bandwidth settings into account.

For more information about the Resulting Frame Rate parameter, see Section 5.1 on page 31.

## Increasing the Maximum Allowed Line Rate

You may find that you would like to acquire lines at a rate higher than the maximum allowed with the camera's current settings. In this case, you must first determine what factor is most restricting the maximum line rate. The descriptions of the three factors that appear below will let you determine which factor is restricting the rate.

### Factor 1:

Factor 1 is the sensor readout time. The readout time for a particular sensor is a fixed value and thus the maximum line acquisition rate as determined by the readout time is also fixed. The table below shows the maximum line rate (in lines per second) based on sensor readout time for each camera model.

| Max Lines/s (based on sensor readout) | |
|---|---|
| **raL2048-48gm** | **raL4096-24gm** |
| 80000 | 80000 |

**Factor 2:**

Factor 2 is the exposure time. You can use the formula below to calculate the maximum line rate based on the exposure time for each acquired line:

$$\text{Max Lines/s} = \frac{1}{\text{Exposure time in µs} + C_1}$$

Where the constant $C_1$ depends on the camera model as shown in the table below:

| | raL2048-48gm | raL4096-24gm |
|---|---|---|
| $C_1$ | 2.6 µs | 2.6 µs |

For more information about setting the exposure time, see Section 8.2.5.2 on page 84.

**Factor 3:**

Factor 3 is the frame transmission time. You can use the formula below to calculate the maximum line rate based on the frame transmission time:

$$\text{Max Lines/s} = \left(\frac{\text{Device Current Throughput Parameter Value}}{\text{Payload Size Parameter Value}}\right) \times \text{Frame Height}$$

Once you have determined which factor is most restrictive on the line rate, you can try to make that factor less restrictive if possible:

- If you find that the sensor readout time is most restrictive factor, you cannot make any adjustments that will result in a higher maximum line rate.
- If you are using long exposure times, it is quite possible to find that your exposure time is the most restrictive factor on the line rate. In this case, you should lower your exposure time. (You may need to compensate for a lower exposure time by using a brighter light source or increasing the opening of your lens aperture.)
- The frame transmission time will not normally be a restricting factor. But if you are using multiple cameras and you have set a small packet size or a large inter-packet delay, you may find that the transmission time is restricting the maximum allowed line rate. In this case, you

could increase the packet size or decrease the inter-packet delay. If you are using several cameras connected to the host PC via a network switch, you could also use a multiport network adapter in the PC instead of a switch. This would allow you to increase the Ethernet bandwidth assigned to the camera and thus decrease the transmission time.

For more information on the settings that determine the bandwidth assigned to the camera, see Section 5.2 on page 33.

## Example

Assume that you are using an raL2048-48gm camera set for an exposure time of 190 µs and a frame height of 500 lines. Also assume that you have checked the value of the Device Current Throughput parameter and the Payload Size parameters and found them to be 110000000 and 5120000 respectively.

**Factor 1 (sensor readout):**

Max Lines/s = 80000

**Factor 2 (exposure time):**

$$\text{Max Lines/s} = \frac{1}{190\ \mu s + 2.6\ \mu s}$$

Max Lines/s = 5192 Lines/s

**Factor 3 (frame transmission time):**

$$\text{Max Lines/s} = \left(\frac{110000000}{5120000}\right) \times 500$$

Max Lines/s = 10742

Factor 2, the exposure time, is the most restrictive factor. In this case, the exposure time setting is limiting the maximum allowed line rate to 5192 lines per second. If you wanted to operate the camera at a higher line rate, you would need to lower the exposure time.

# 9 Pixel Data Formats

By selecting a pixel data format, you determine the format (layout) of the image data transmitted by the camera. This section provides detailed information about the available pixel data formats.

## 9.1 Setting the Pixel Data Format

The setting for the camera's Pixel Format parameter determines the format of the pixel data that will be output from the camera. Table 8 lists the pixel formats available on each camera type.

| Mono Camera Pixel Formats |
|---|
| Mono 8 |
| Mono 12 |
| Mono 12 Packed |
| YUV 4:2:2 Packed |
| YUV 4:2:2 (YUYV) Packed |

Table 8: Available Pixel Formats

Details of the monochrome camera formats are described in Section 9.2 on page 124.

You can set the Pixel Format parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to set the parameter value:

```
Camera.PixelFormat.SetValue( PixelFormat_Mono8 );
Camera.PixelFormat.SetValue( PixelFormat_Mono12 );
Camera.PixelFormat.SetValue( PixelFormat_Mono12Packed );
Camera.PixelFormat.SetValue( PixelFormat_YUV422Packed );
Camera.PixelFormat.SetValue( PixelFormat_YUV422_YUYV_Packed );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

# 9.2 Pixel Data Formats

## 9.2.1 Mono 8 Format

When a monochrome camera is set for the Mono 8 pixel data format, it outputs 8 bits of brightness data per pixel.

The table below describes how the pixel data for a received frame will be ordered in the image buffer in your PC when the camera is set for Mono 8 output.

The following standards are used in the table:

$P_0$ = the first pixel transmitted by the camera

$P_n$ = the last pixel transmitted by the camera

$B_0$ = the first byte in the buffer

$B_m$ = the last byte in the buffer

| Byte | Pixel - Data Bits |
|---|---|
| $B_0$ | $P_0$ 7 ... 0 |
| $B_1$ | $P_1$ 7 ... 0 |
| $B_2$ | $P_2$ 7 ... 0 |
| $B_3$ | $P_3$ 7 ... 0 |
| $B_4$ | $P_4$ 7 ... 0 |
| • | • |
| • | • |

| Byte | Pixel - Data bits |
|---|---|
| • | • |
| • | • |
| $B_{m-4}$ | $P_{n-4}$ 7 ... 0 |
| $B_{m-3}$ | $P_{n-3}$ 7 ... 0 |
| $B_{m-2}$ | $P_{n-2}$ 7 ... 0 |
| $B_{m-1}$ | $P_{n-1}$ 7 ... 0 |
| $B_m$ | $P_n$ 7 ... 0 |

With the camera set for Mono 8, the pixel data output is 8 bit data of the “unsigned char” type. The available range of data values and the corresponding indicated signal levels are as shown in the table below.

| This Data Value (Hexadecimal) | Indicates This Signal Level (Decimal) |
|---|---|
| 0xFF | 255 |
| 0xFE | 254 |
| • | • |
| • | • |
| • | • |
| 0x01 | 1 |
| 0x00 | 0 |

## 9.2.2 Mono 12 Format

When a monochrome camera is set for the Mono 12 pixel data format, it outputs 16 bits of brightness data per pixel with 12 bits effective. The 12 bits of effective pixel data fill from the least significant bit. The four unused most significant bits are filled with zeros.

The table below describes how the pixel data for a received frame will be ordered in the image buffer in your PC when the camera is set for Mono 12 output. Note that the data is placed in the image buffer in **little endian format**.

The following standards are used in the table:

$P_0$ = the first pixel transmitted by the camera

$P_n$ = the last pixel transmitted by the camera

$B_0$ = the first byte in the buffer

$B_m$ = the last byte in the buffer

x = unused bit, zero filled

| **Byte** | **Pixel - Data Bits** |
|---|---|
| $B_0$ | $P_0$ 7 ... 0 |
| $B_1$ | $P_0$ x x x x 11 ... 8 |
| $B_2$ | $P_1$ 7 ... 0 |
| $B_3$ | $P_1$ x x x x 11 ... 8 |
| $B_4$ | $P_2$ 7 ... 0 |
| $B_5$ | $P_2$ x x x x 11 ... 8 |
| $B_6$ | $P_3$ 7 ... 0 |
| $B_7$ | $P_3$ x x x x 11 ... 8 |
| $B_8$ | $P_4$ 7 ... 0 |
| $B_9$ | $P_4$ x x x x 11 ... 8 |
| • | • |
| • | • |
| • | • |
| $B_{m-7}$ | $P_{n-3}$ 7 ... 0 |
| $B_{m-6}$ | $P_{n-3}$ x x x x 11 ... 8 |
| $B_{m-5}$ | $P_{n-2}$ 7 ... 0 |
| $B_{m-4}$ | $P_{n-2}$ x x x x 11 ... 8 |
| $B_{m-3}$ | $P_{n-1}$ 7 ... 0 |
| $B_{m-2}$ | $P_{n-1}$ x x x x 11 ... 8 |
| $B_{m-1}$ | $P_n$ 7 ... 0 |
| $B_m$ | $P_n$ x x x x 11 ... 8 |

When the camera is set for Mono 12, the pixel data output is 16 bit data of the “unsigned short (little endian)” type. The available range of data values and the corresponding indicated signal levels are as shown in the table below. Note that for 16 bit data, you might expect a value range from 0x0000 to 0xFFFF. However, with the camera set for Mono 12 only 12 bits of the 16 bits transmitted are effective. Therefore, the highest data value you will see is 0x0FFF indicating a signal level of 4095.

| This Data Value (Hexadecimal) | Indicates This Signal Level (Decimal) |
|---|---|
| 0x0FFF | 4095 |
| 0x0FFE | 4094 |
| • | • |
| • | • |
| • | • |
| 0x0001 | 1 |
| 0x0000 | 0 |

## 9.2.3 Mono 12 Packed Format

When a monochrome camera is set for the Mono 12 Packed pixel data format, it outputs 12 bits of brightness data per pixel. Every three bytes transmitted by the camera contain data for two pixels.

The table below describes how the pixel data for a received frame will be ordered in the image buffer in your PC when the camera is set for Mono 12 Packed output.

The following standards are used in the table:

$P_0$ = the first pixel transmitted by the camera

$P_n$ = the last pixel transmitted by the camera

$B_0$ = the first byte in the buffer

$B_m$ = the last byte in the buffer

| **Byte** | **Pixel - Data Bits** | |
|---|---|---|
| $B_0$ | $P_0$ 11 ... 4 | |
| $B_1$ | $P_1$ 3 ... 0 | $P_0$ 3 ... 0 |
| $B_2$ | $P_1$ 11 ... 4 | |
| $B_3$ | $P_2$ 11 ... 4 | |
| $B_4$ | $P_3$ 3 ... 0 | $P_2$ 3 ... 0 |
| $B_5$ | $P_3$ 11 ... 4 | |
| $B_6$ | $P_4$ 11 ... 4 | |
| $B_7$ | $P_5$ 3 ... 0 | $P_4$ 3 ... 0 |
| $B_8$ | $P_5$ 11 ... 4 | |
| $B_9$ | $P_6$ 11 ... 4 | |
| $B_{10}$ | $P_7$ 3 ... 0 | $P_6$ 3 ... 0 |
| $B_{11}$ | $P_7$ 1 ... 4 | |
| • | • | |
| • | • | • |
| • | • | |
| $B_{m-5}$ | $P_{n-3}$ 11 ... 4 | |
| $B_{m-4}$ | $P_{n-2}$ 3 ... 0 | $P_{n-3}$ 3 ... 0 |
| $B_{m-3}$ | $P_{n-2}$ 11 ... 4 | |
| $B_{m-2}$ | $P_{n-1}$ 11 ... 4 | |
| $B_{m-1}$ | $P_n$ 3 ... 0 | $P_{n-1}$ 3 ... 0 |
| $B_m$ | $P_n$ 11 ... 4 | |

When a monochrome camera is set for Mono 12 Packed, the pixel data output is 12 bit data of the “unsigned” type. The available range of data values and the corresponding indicated signal levels are as shown in the table below.

| This Data Value (Hexadecimal) | Indicates This Signal Level (Decimal) |
|---|---|
| 0x0FFF | 4095 |
| 0x0FFE | 4094 |
| • | • |
| • | • |
| • | • |
| 0x0001 | 1 |
| 0x0000 | 0 |

## 9.2.4 YUV 4:2:2 Packed Format

When a monochrome camera is set for the YUV 4:2:2 Packed pixel data format, the camera transmits Y, U, and V values in a fashion that mimics the output from a color camera set for YUV 4:2:2 Packed.

The Y value transmitted for each pixel is the actual 8 bit brightness value similar to the pixel data transmitted when a monochrome camera is set for Mono 8. The U and V values transmitted will always be zero. With this format, a Y value is transmitted for each pixel, but the U and V values are only transmitted for every second pixel.

The table below describes how the pixel data for a received frame will be ordered in the image buffer in your PC when the camera is set for YUV 4:2:2 Packed output.

The following standards are used in the table:

$P_0$ = the first pixel transmitted by the camera

$P_n$ = the last pixel transmitted by the camera

$B_0$ = the first byte in the buffer

$B_m$ = the last byte in the buffer

| **Byte** | **Pixel - Data Bits** |
|---|---|
| $B_0$ | U $P_0$ 7 ... 0 |
| $B_1$ | Y $P_0$ 7 ... 0 |
| $B_2$ | V $P_0$ 7 ... 0 |
| $B_3$ | Y $P_1$ 7 ... 0 |
| $B_4$ | U $P_2$ 7 ... 0 |
| $B_5$ | Y $P_2$ 7 ... 0 |
| $B_6$ | V $P_2$ 7 ... 0 |
| $B_7$ | Y $P_3$ 7 ... 0 |
| $B_8$ | U $P_4$ 7 ... 0 |
| $B_9$ | Y $P_4$ 7 ... 0 |
| $B_{10}$ | V $P_4$ 7 ... 0 |
| $B_{11}$ | Y $P_5$ 7 ... 0 |
| • | • |
| • | • |
| • | • |
| $B_{m-7}$ | U $P_{n-3}$ 7 ... 0 |
| $B_{m-6}$ | Y $P_{n-3}$ 7 ... 0 |
| $B_{m-5}$ | V $P_{n-3}$ 7 ... 0 |
| $B_{m-4}$ | Y $P_{n-2}$ 7 ... 0 |
| $B_{m-3}$ | U $P_{n-1}$ 7 ... 0 |

| | | |
|---|---|---|
| $B_{m-2}$ | Y $P_{n-1}$ | 7 ... 0 |
| $B_{m-1}$ | V $P_{n-1}$ | 7 ... 0 |
| $B_m$ | Y $P_n$ | 7 ... 0 |

When the camera is set for YUV 4:2:2 Packed output, the pixel data output for the Y component is 8 bit data of the “unsigned char” type. The range of data values for the Y component and the corresponding indicated signal levels are shown below.

| **This Data Value (Hexadecimal)** | **Indicates This Signal Level (Decimal)** |
|---|---|
| 0xFF | 255 |
| 0xFE | 254 |
| • | • |
| • | • |
| • | • |
| 0x01 | 1 |
| 0x00 | 0 |

The pixel data output for the U component or the V component is 8 bit data of the “straight binary” type and will always be zero.

| **This Data Value (Hexadecimal)** | **Indicates This Signal Level (Decimal)** |
|---|---|
| 0x00 | 0 |

## 9.2.5 YUV 4:2:2 (YUYV Packed) Format

When a monochrome camera is set for the YUV 4:2:2 (YUYV) Packed pixel data format, the camera transmits Y, U, and V values in a fashion that mimics the output from a color camera set for YUV 4:2:2 (YUYV) Packed.

The YUV 4:2:2 (YUYV) packed pixel data format is similar to the YUV 4:2:2 pixel format described in the previous section. The only difference is the order of the bytes transmitted to the host PC. With the YUV 4:2:2 format, the bytes are ordered as specified in the DCAM standard issued by the 1394 Trade Association. With the YUV 4:2:2 (YUYV) format, the bytes are ordered to emulate the ordering normally associated with analog frame grabbers and Windows® frame buffers.

The Y value transmitted for each pixel is the actual 8 bit brightness value similar to the pixel data transmitted when a monochrome camera is set for Mono 8. The U and V values transmitted will always be zero. With this format, a Y value is transmitted for each pixel, but the U and V values are only transmitted for every second pixel.

The table below describes how the pixel data for a received frame will be ordered in the image buffer in your PC when the camera is set for YUV 4:2:2 (YUYV) output.

With this format, the Y component is transmitted for each pixel, but the U and V components are only transmitted for every second pixel.

The following standards are used in the table:

$P_0$ = the first pixel transmitted by the camera

$P_n$ = the last pixel transmitted by the camera

$B_0$ = the first byte in the buffer

$B_m$ = the last byte in the buffer

| **Byte** | **Pixel - Data Bits** |
|---|---|
| $B_0$ | Y $P_0$ 7 ... 0 |
| $B_1$ | U $P_0$ 7 ... 0 |
| $B_2$ | Y $P_1$ 7 ... 0 |
| $B_3$ | V $P_0$ 7 ... 0 |
| $B_4$ | Y $P_2$ 7 ... 0 |
| $B_5$ | U $P_2$ 7 ... 0 |
| $B_6$ | Y $P_3$ 7 ... 0 |
| $B_7$ | V $P_2$ 7 ... 0 |
| $B_8$ | Y $P_4$ 7 ... 0 |
| $B_9$ | U $P_4$ 7 ... 0 |
| $B_{10}$ | Y $P_5$ 7 ... 0 |
| $B_{11}$ | V $P_4$ 7 ... 0 |
| • | • |
| • | • |
| • | • |

| | |
|---|---|
| $B_{m-7}$ | Y $P_{n-3}$ 7 ... 0 |
| $B_{m-6}$ | U $P_{n-3}$ 7 ... 0 |
| $B_{m-5}$ | Y $P_{n-2}$ 7 ... 0 |
| $B_{m-4}$ | V $P_{n-3}$ 7 ... 0 |
| $B_{m-3}$ | Y $P_{n-1}$ 7 ... 0 |
| $B_{m-2}$ | U $P_{n-1}$ 7 ... 0 |
| $B_{m-1}$ | Y $P_{n}$ 7 ... 0 |
| $B_{m}$ | V $P_{n-1}$ 7 ... 0 |

When the camera is set for YUV 4:2:2 (YUYV) output, the pixel data output for the Y component is 8 bit data of the “unsigned char” type. The range of data values for the Y component and the corresponding indicated signal levels are shown below.

| **This Data Value (Hexadecimal)** | **Indicates This Signal Level (Decimal)** |
|---|---|
| 0xFF | 255 |
| 0xFE | 254 |
| • | • |
| • | • |
| • | • |
| 0x01 | 1 |
| 0x00 | 0 |

The pixel data output for the U component or the V component is 8 bit data of the “straight binary” type and will always be zero.

| **This Data Value (Hexadecimal)** | **Indicates This Signal Level (Decimal)** |
|---|---|
| 0x00 | 0 |

# 9.3 Pixel Transmission Sequence

For each acquired frame, pixel data is transmitted from the camera in the following sequence:

| | | | | | | |
|---|---|---|---|---|---|---|
| $Row_0 Col_0$, | $Row_0 Col_1$, | $Row_0 Col_2$ | .. .. | $Row_0 Col_{m-2}$, | $Row_0 Col_{m-1}$, | $Row_0 Col_m$ |
| $Row_1 Col_0$, | $Row_1 Col_1$, | $Row_1 Col_2$ | .. .. | $Row_1 Col_{m-2}$, | $Row_1 Col_{m-1}$, | $Row_1 Col_m$ |
| $Row_2 Col_0$, | $Row_2 Col_1$, | $Row_2 Col_2$ | .. .. | $Row_2 Col_{m-2}$, | $Row_2 Col_{m-1}$, | $Row_2 Col_m$ |
| : | : | : | | : | : | : |
| : | : | : | | : | : | : |
| $Row_{n-2} Col_0$, | $Row_{n-2} Col_1$, | $Row_{n-2} Col_2$ | .. .. | $Row_{n-2} Col_{m-2}$, | $Row_{n-2} Col_{m-1}$, | $Row_{n-2} Col_m$ |
| $Row_{n-1} Col_0$, | $Row_{n-1} Col_1$, | $Row_{n-1} Col_2$ | .. .. | $Row_{n-1} Col_{m-2}$, | $Row_{n-1} Col_{m-1}$, | $Row_{n-1} Col_m$ |
| $Row_n Col_0$, | $Row_n Col_1$, | $Row_n Col_2$ | .. .. | $Row_n Col_{m-2}$, | $Row_n Col_{m-1}$, | $Row_n Col_m$ |

Where:

- $Row_0 Col_0$ is the upper left corner of the frame
- The columns are numbered 0 through m from the left side to the right side of the frame
- The rows are numbered 0 through n from the top to the bottom of the frame, corresponding to n+1 line acquisitions.

# 10 Standard Features

This chapter provides detailed information about the standard features available on each camera. It also includes an explanation of their operation and the parameters associated with each feature.

## 10.1 Gain and Black Level

### 10.1.1 Gain

The camera's gain is adjustable. As shown in Fig. 38, increasing the gain increases the slope of the response curve for the camera. This results in an increase in the gray values output from the camera for a given amount of output from the imaging sensor. Decreasing the gain decreases the slope of the response curve and results in lower gray values for a given amount of sensor output.

Increasing the gain is useful when at your brightest exposure, the highest gray values achieved are lower than 255 (for pixel data formats with 8 bit depth) or 4095 (for pixel data formats with 12 bit depth). For example, if you found that at your brightest exposure the gray values output by the camera were no higher than 127 (in an 8 bit format), you could increase the gain to 6 dB (an amplification factor of 2) and thus reach gray values of 254.

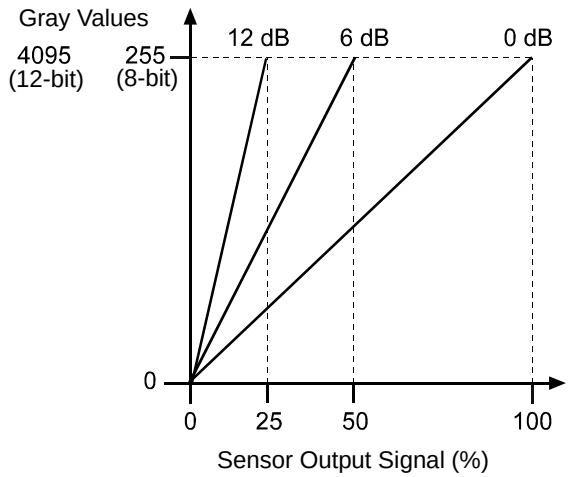



Fig. 38: Gain in dB

You can use the analog gain for coarsely setting gain and the digital gain for finer adjustment.

### 10.1.1.1 Analog Gain

The camera's analog gain is determined by the Gain parameter with the gain selector set to Analog All. All pixels in the sensor are affected by this setting.

The allowed parameter values are 1 and 4. A parameter value of 1 corresponds to 0 dB and gain will not be modified. A parameter value of 4 corresponds to 12 dB and an amplification factor of 4.

#### Setting the Analog Gain

You must stop image acquisition by issuing an acquisition stop command before changing the analog gain settings.

For more information about the acquisition stop command, see Section 8.2.1 on page 68.

To set the Gain Analog All parameter value:

- Set the Gain Selector to Analog All.
- Set the Gain parameter to 1 or 4, as desired.

You can set the Gain Selector and the Gain parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Set Gain Analog All
Camera.GainSelector.SetValue( GainSelector_AnalogAll );
Camera.Gain.Raw.SetValue( 4 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

### 10.1.1.2 Digital Gain

Adjusting the camera's digital gain will digitally shift the group of bits that is output for the pixel values from each ADC in the camera.

Increasing the digital gain parameter value will result in an amplified gain and therefore in higher pixel values. Decreasing the digital gain setting will result in a decreased gain and therefore in lower pixel values.

The digital gain parameter values can be set on an integer scale ranging from 264 to 2047. This range of settings is linearly related to a range of amplification factors where a parameter value of

264 corresponds to 0 dB and gain will not be modified and a parameter value of 2047 corresponds to 17.790 dB and an amplification factor of approximately 7.75.

You can use the formula below to calculate the dB of gain that will result from the Gain Raw parameter values:

$$Gain[dB] = 20 \times \log_{10}\left(\frac{\text{Gain Raw}}{264}\right)$$

$$Gain[dB] = 20 \times \log_{10}(\text{Gain Raw}) - 48.432$$

Note that due to the nature of digital gain certain gray values will be absent in the image ("missing codes") if digital gain is set to a value larger than 264.

You can use the remove parameter limits feature to remove to lower limit for digital gain parameter values. When you use the remove parameter limits feature you can also set digital gain parameter values in the range from 0 to 263. This corresponds to a range of amplification factors from 0 to approximately 0.99.

If the digital gain parameter value is set below 264 using the remove parameter limits feature: In this case, regardless of the brightness of illumination, the camera will not be able to reach the maximum gray values that otherwise could be reached. For example, if the camera is set to a 12 bit pixel data format, the maximum gray value of 4095 can not be reached if the digital gain parameter value is set below 264.

For more information about the remove parameter limits feature, see Section 10.2 on page 140.

## Setting the Digital Gain

To set the Gain All parameter value:

- Set the Gain Selector to All.
- Set the Gain parameter to your desired value.

You can set the Gain Selector and the Gain parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Set Gain Digital All
Camera.GainSelector.SetValue( GainSelector_All );
Camera.GainRaw.SetValue( 264 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

#### 10.1.1.3 Using Both Analog Gain and Digital Gain

You can use analog gain and digital gain at the same time. In this case, the amplification factors will multiply. For example, if you set analog gain to an amplification factor of 4 and use an amplification factor of 1.2 for digital gain the total amplification factor will be 4.8. This corresponds to adding 12 dB and 1.6 dB to give a total gain of 13.6 dB.

For optimum image quality, we recommend to set the total amplification as low as possible. If you need an amplification factor larger than 4 we recommend to set analog gain to 4 and then digital gain to reach the desired total amplification.

If you use analog gain and digital gain at the same time and also use the remove parameter limits for digital gain with digital gain parameter values below 264: Note also that the amplification factor for total gain will be 0 if the digital gain setting is 0.

## 10.1.2 Black Level

Adjusting the camera's black level will result in an offset to the pixel values output from the camera.

The camera's black level is determined by the Black Level Raw parameter with the black Level selector set to All. All pixels in the sensor are affected by this setting.

If the camera is set for a pixel data format with an 8 bit depth, an increase of 16 in a black level setting will result in a positive offset of 1 in the pixel values output from the camera. And a decrease of 16 in a black level setting will result in a negative offset of 1 in the pixel values output from the camera.

If the camera is set for a pixel data format with a 12 bit depth, an increase of 1 in a black level setting will result in a positive offset of 1 in the pixel values output from the camera. A decrease of 1 in a black level setting will result in a negative offset of 1 in the pixel values output from the camera.

### Setting the Black Level

To set the Black Level Raw All value:

- Set the Black Level Selector to All.
- Set the Black Level Raw parameter to your desired value.

You can set the Black Level Selector and the Black Level Raw parameter values from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Set Black Level Raw All
Camera.BlackLevelSelector.SetValue ( BlackLevelSelector_All );
Camera.BlackLevelRaw.SetValue( 64 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

# 10.2 Remove Parameter Limits

For each camera feature, the allowed range of any associated parameter values is normally limited. The factory limits are designed to ensure optimum camera operation and, in particular, good image quality. For special camera uses, however, it may be helpful to set parameter values outside of the factory limits.

The remove parameter limits feature lets you remove the factory limits for parameters associated with certain camera features. When the factory limits are removed, the parameter values can be set within extended limits. Typically, the range of the extended limits is dictated by the physical restrictions of the camera's electronic devices, such as the absolute limits of the camera's variable gain control.

The values for any extended limits can be determined by using the Basler pylon Viewer or from within your application via the pylon API.

Currently, the limits can be removed from the digital gain feature. Removing the parameter limits on the digital gain feature will only remove the lower limit. The lower limit for the DigitalGain parameter is reduced to 0.

For more information about the digital gain feature, see Section 10.1 on page 135.

### Removing Parameter Limits

To remove the limits for a parameter:

- Use the Parameter Selector to select the parameter whose limits you want to remove.
- Set the value of the Remove Limits parameter.

You can set the Parameter Selector and the value of the Remove Limits parameter from within your application software by using the Basler pylon API. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Select the feature whose factory limits will be removed.
Camera.ParameterSelector.SetValue( ParameterSelector_Gain );
// Remove the limits for the selected feature.
Camera.RemoveLimits.SetValue( true );
```

You can also use the Basler pylon Viewer application to easily set the parameters. Note that the remove parameter limits feature will only be available at the "guru" viewing level.

# 10.3 Event Reporting

Event reporting is available on the camera. With event reporting, the camera can generate an "event" and after some intermediate steps transmit a related event message to the PC whenever a specific situation has occurred.

The camera can generate and transmit events for the following types of situations:

- Overtriggering of the acquisition start trigger has occurred (AcquisitionStartOvertriggerEventData).
  This happens if the camera receives an acquisition start trigger while it is currently not in the waiting for an acquisition start trigger status.
- Overtriggering of the frame start trigger has occurred (FrameStartOvertriggerEventData).
  This happens if the camera receives a frame start trigger while it is currently not in the waiting for a frame start trigger status.
- Overtriggering of the line start trigger has occurred (LineStartOvertriggerEventData).
  This happens if the camera receives a line start trigger while it is currently in the process of acquiring a line.
- A frame timeout has occurred (FrameTimeoutEventData).
  This happens if the acquisition of a frame is not completed within a set period, provided frame timeout is enabled and configured. For more information, see the Frame Timeout section.
- An event overrun has occurred (EventOverrunEventData).
  This situation is explained later in this section.

### An Example of Event Reporting

An example related to the Frame Start Overtrigger event illustrates how event reporting works. The example assumes that your system is set for event reporting (see below) and that the camera has received a frame start trigger while it is currently in the process of acquiring a frame. In this case:

1. A Frame Start Overtrigger event is created. The event contains the event in the strict sense and supplementary information:

   An *Event Type Identifier*. In this case, the identifier would show that a frame start overtrigger type event has occurred.

   A *Stream Channel Identifier*. Currently this identifier is always 0.

   A *Timestamp*. This is a timestamp indicating when the event occurred. (The time stamp timer starts running at power off/on or at camera reset. The unit for the timer is "ticks" where one tick = 8 ns. The timestamp is a 64 bit value.)
2. The event is placed in an internal queue in the camera.
3. As soon as network transmission time is available, an event message will be sent to the PC. If only one event is in the queue, the message will contain the single event. If more than one event is in the queue, the message will contain multiple events.
   a. After the camera sends an event message, it waits for an acknowledgement. If no acknowledgement is received within a specified timeout, the camera will resend the event message. If an acknowledgement is still not received, the timeout and resend mechanism will repeat until a specified maximum number of retries is reached. If the maximum number of retries is reached and no acknowledge has been received, the message will be dropped.

During the time that the camera is waiting for an acknowledgement, no new event messages can be transmitted.

4. Event reporting involves some further software-related steps and settings to be made. For more information, see the "Camera Events" code sample included with the pylon software development kit.

## The Event Queue

As mentioned in the example above, the camera has an event queue. The intention of the queue is to handle short term delays in the camera's ability to access the network and send event messages. When event reporting is working "smoothly", a single event will be placed in the queue and this event will be sent to the PC in an event message before the next event is placed in the queue. If there is an occasional short term delay in event message transmission, the queue can buffer several events and can send them within a single event message as soon as transmission time is available.

However, if you are operating the camera at high line rates, the camera may be able to generate and queue events faster than they can be transmitted and acknowledged. In this case:

1. The queue will fill and events will be dropped.
2. An event overrun will occur.
3. Assuming that you have event overrun reporting enabled, the camera will generate an "event overrun event" and place it in the queue.
4. As soon as transmission time is available, an event message containing the event overrun event will be transmitted to the PC.

The event overrun event is simply a warning that events are being dropped. The notification contains no specific information about how many or which events have been dropped.

## Setting Your System for Event Reporting

Event reporting must be enabled in the camera and some additional software-related settings must be made. This is described in the "Camera Events" code sample included with the pylon software development kit.

Event reporting must be specifically set up for each type of event using the parameter name of the event and of the supplementary information. The following table lists the relevant parameter names:

| Event | Event Parameter Name | Supplementary Information Parameter Name |
|---|---|---|
| Acquisition Start Overtrigger | AcquisitionStartOvertriggerEventData | AcquisitionStartOvertriggerEventStreamChannelIndex |
| | | AcquisitionStartOvertriggerEventTimestamp |
| Frame Start Overtrigger | FrameStartOvertriggerEventData | FrameStartOvertriggerEventStreamChannelIndex |
| | | FrameStartOvertriggerEventTimestamp |
| Line Start Overtrigger | LineStartOvertriggerEventData | LineStartOvertriggerEventChannelIndex |
| | | LineStartOvertriggerEventTimestamp |
| Frame Timeout | FrameTimeoutEventData | FrameTimeoutEventStreamChannelIndex |
| | | FrameTimeoutEventTimestamp |
| Event Overrun | EventOverrunEventData | EventOverrunEventStreamChannelIndex |
| | | EventOverrunEventTimestamp |

Table 9: Parameter Names of Events and Supplementary Information

You can enable event reporting and make the additional settings from within your application software by using the pylon API. The pylon software development kit includes a "Camera Events" code sample that illustrates the entire process.

For more detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

# 10.4 Luminance Lookup Table

The type of electronics used on the camera allow the camera's sensor to acquire pixel values at a 12 bit depth. Normally, when a camera is set for a 12 bit pixel data format, the camera uses the actual 12 bit pixel values reported by the sensor.

The luminance lookup table feature lets you create a custom 12 bit to 12 bit lookup table that maps the actual 12 bit values output from the sensor to substitute 12 bit values of your choice. When the lookup table is enabled, the camera will replace the actual pixel values output from the sensor with the substitute values from the table.

The lookup table has 4096 indexed locations with a 12 bit value stored at each index. The values stored in the table are used like this:

- When the sensor reports that a pixel has an actual 12 bit value of 0, the substitute 12 bit value stored at index 0 will replace the actual pixel value.
- The numbers stored at indices 1 through 7 are not used.
- When the sensor reports that a pixel has an actual 12 bit value of 8, the substitute 12 bit value stored at index 8 will replace the actual pixel value.
- The numbers stored at indices 9 through 15 are not used.
- When the sensor reports that a pixel has an actual 12 bit value of 16, the substitute 12 bit value stored at index 16 will replace the actual pixel value.
- The numbers stored at indices 17 through 23 are not used.
- When the sensor reports that a pixel has an actual 12 bit value of 24, the substitute 12 bit value stored at index 24 will replace the actual pixel value.
- And so on.

As you can see, the table does not include a defined 12 bit substitute value for every actual pixel value that the sensor can report. If the sensor reports an actual pixel value that is between two values that have a defined substitute, the camera performs a straight line interpolation to determine the substitute value that it should use. For example, assume that the sensor reports an actual pixel value of 12. In this case, the camera would perform a straight line interpolation between the substitute values at index 8 and index 16 in the table. The result of the interpolation would be used by the camera as the substitute.

Another thing to keep in mind about the table is that index 4088 is the last index that will have a defined substitute value associated with it (the values at indices 4089 through 4095 are not used.) If the sensor reports an actual value greater than 4088, the camera will not be able to perform an interpolation. In cases where the sensor reports an actual value greater than 4088, the camera simply uses the 12 bit substitute value from index 4088 in the table.

The advantage of the luminance lookup table feature is that it lets a user customize the response curve of the camera. The graphs below represent the contents of two typical lookup tables. The first graph is for a lookup table where the values are arranged so that the output of the camera increases linearly as the actual sensor output increases. The second graph is for a lookup table where the values are arranged so that the camera output increases quickly as the actual sensor output moves from 0 through 2048 and increases gradually as the actual sensor output moves from 2049 through 4096.

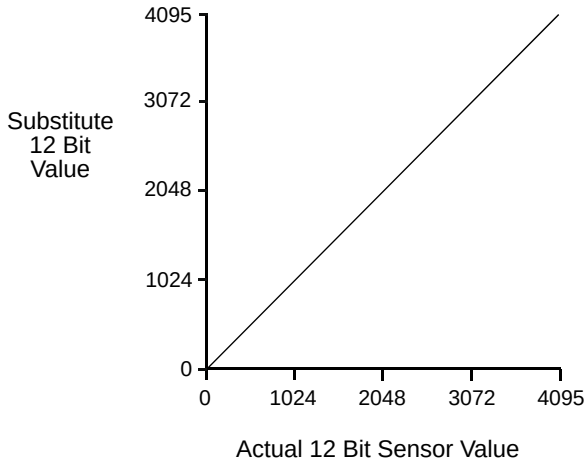



Fig. 39: Lookup Table with Values Mapped in a Linear Fashion

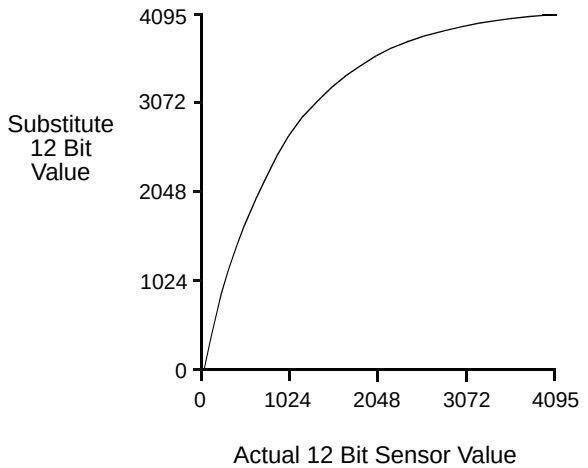



Fig. 40: Lookup Table with Values Mapped for Higher Camera Output at Low Sensor Readings

## Using the Luminance Lookup Table to Get 8 Bit Output

As mentioned above, when the camera is set for a 12 bit pixel data format, the lookup table can be used to perform a 12 bit to 12 bit substitution. The lookup table can also be used in 12 bit to 8 bit fashion. To use the table in 12 bit to 8 bit fashion, you enter 12 bit substitution values into the table and enable the table as you normally would. But instead of setting the camera for a 12 bit pixel data format, you set the camera for an 8 bit format (such as Mono 8). In this situation, the camera will first use the values in the table to do a 12 bit to 12 bit substitution. It will then truncate the lowest 4 bits of the substitute value and will transmit the remaining 8 highest bits.

### Changing the Values in the Luminance Lookup Table and Enabling the Table

You can change the values in the luminance lookup table (LUT) and enable the use of the lookup table by doing the following:

1. Use the LUT Selector to select a lookup table. (Currently there is only one lookup table available, i.e., the "luminance" lookup table described above.)
2. Use the LUT Index parameter to select an index number.
3. Use the LUT Value parameter to enter the substitute value that will be stored at the index number that you selected in step 2.
4. Repeat steps 2 and 3 to enter other substitute values into the table as desired.
5. Use the LUT Enable parameter to enable the table.

You can set the LUT Selector, the LUT Index parameter and the LUT Value parameter from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and the parameter values:

```
// Select the lookup table
Camera.LUTSelector.SetValue( LUTSelector_Luminance );

// Write a lookup table to the device.
// The following lookup table causes an inversion of the sensor values
// ( bright -> dark, dark -> bright )
for ( int i = 0; i < 4096; i += 8 )
{
   Camera.LUTIndex.SetValue( i );
   Camera.LUTValue.SetValue( 4095 - i );
}
// Enable the lookup table
Camera.LUTEnable.SetValue( true );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

# 10.5 Binning

Binning increases the camera's response to light by summing the charges from adjacent pixels into one pixel.

With horizontal binning, the charges of 2, 3, or a maximum of 4 adjacent pixels are summed and are reported out of the camera as a single pixel. Fig. 41 illustrates horizontal binning.

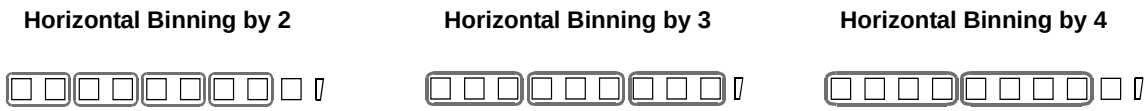



Fig. 41: Horizontal Binning

## Setting Binning

You can enable horizontal binning by setting the Binning Horizontal parameter. Setting the parameter's value to 2, 3, or 4 enables horizontal binning by 2, horizontal binning by 3, or horizontal binning by 4 respectively. Setting the parameter's value to 1 disables horizontal binning.

You can set the Binning Vertical or the Binning Horizontal parameter value from within your application software by using the Basler pylon API. The following code snippet illustrates using the API to set the parameter values:

```
// Enable horizontal binning by 4
Camera.BinningHorizontal.SetValue( 4 );

// Disable horizontal binning
Camera.BinningHorizontal.SetValue( 1 );
```

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon API and the pylon Viewer, see Section 3 on page 31.

# 10.6 Gamma Correction

The gamma correction feature lets you modify the brightness of the pixel values output by the camera's sensor to account for a non-linearity in the human perception of brightness. To accomplish the correction, a gamma correction factor ($\gamma$) is applied to the brightness value (Y) of each pixel according to the following formula:

$$Y_{corrected} = \left(\frac{Y_{uncorrected}}{Y_{max}}\right)^{\gamma} \times Y_{max}$$

The formula uses uncorrected and corrected pixel brightnesses that are normalized by the maximum pixel brightness. The maximum pixel brightness equals 255 for 8 bit output and 4095 for 12 bit output.

When the gamma correction factor is set to 1, the output pixel brightness will not be corrected.

A gamma correction factor between 0 and 1 will result in increased overall brightness, and a gamma correction factor greater than 1 will result in decreased overall brightness.

In all cases, black (output pixel brightness equals 0) and white (output pixel brightness equals 255 at 8 bit output and 4095 at 12 bit output) will not be corrected.

### Enabling Gamma Correction and Setting the Gamma

You can enable or disable the gamma correction feature by setting the value of the Gamma Enable parameter.

When gamma correction is enabled, the correction factor is determined by the value of the Gamma parameter. The Gamma parameter can be set in a range from 0 to 3.99902. So if the Gamma parameter is set to 1.2, for example, the gamma correction factor will be 1.2.

You can set the Gamma Enable and Gamma parameter values from within your application software by using the Basler pylon API. The following code snippet illustrates using the API to set the parameter values:

```
// Enable the Gamma feature
Camera.GammaEnable.SetValue( true );

// Set the Gamma value to 1.2
Camera.Gamma.SetValue( 1.2 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

# 10.7 Shading Correction

Two types of shading correction are available on the camera, offset shading correction and gain shading correction. You can set the camera to only perform offset shading correction, to only perform gain shading correction, or to perform both types of shading correction.

## 10.7.1 Offset Shading Correction

When a line scan camera with a digital sensor captures a line in complete darkness, all of the pixel values in the line should be near zero and they should be equal. In practice, slight variations in the performance of the pixels in the sensor will cause some variation in the pixel values output from the camera when the camera is capturing lines in darkness. Offset shading correction (also known as dark signal non-uniformity (DSNU) correction) corrects for this type of variation.

Offset shading correction works by adding an individual gray value to each pixel value in the acquired lines. The gray values used for correction are included in a "shading file", commonly also referred to as a "shading set". In order to use offset shading correction, the user must enable offset shading correction and activate the related defaultshading file or the previously created usershading file (see below).

## 10.7.2 Gain Shading Correction

When a line scan camera with a digital sensor captures a line with the camera viewing a uniform light-colored target in bright light, all of the pixel values in the line should be near their maximum gray value and they should be equal. In practice, slight variations in the performance of the pixels in the sensor, variations in the optics, and variations in the lighting will cause some variation in the pixel values output from the camera. Gain shading correction (also known as photo response non-uniformity (PRNU) correction) corrects for this type of variation.

Gain shading correction works by applying an individual multiplier to each pixel value in the acquired lines. The multipliers used for correction are included in a "shading file", commonly also referred to as a "shading set". In order to use gain shading correction, the user must enable gain shading correction and activate the related defaultshading file or the previously created usershading file (see below).

## 10.7.3 Default Shading Set File and User Shading Set File

For each type of shading correction, two types of shading set files are available in the camera's nonvolatile memory:

- The first type of shading set file is called the "defaultshading" file. One "defaultshading" file is available for offset shading correction and another one for gain shading correction.

  A "defaultshading" file contains a complete collection of the values needed to perform either offset shading or gain shading correction. The values in the files are generated during the camera's factory setup procedure and they essentially serve as default shading correction values. The values are optimized for performing shading correction with "standard" optics and lighting. Using the factory values will give you reasonable shading correction performance in most situations. One advantage of the factory values is that they serve as a good default. The "defaultshading" files are in a protected area of the camera's memory and can't be changed.
- The second type of shading set file is called the "usershading" file. One "usershading" file is available for offset shading correction and another one for gain shading correction.

  Like a "defaultshading" file, a "usershading" file also holds a complete collection of the values needed to perform either offset shading or gain shading correction. The values stored in the files must, however, be generated by the camera user. When the values are generated the camera must operate under its real world conditions. The "usershading" files contain the shading correction values that will normally be used for day-to-day camera operation. A procedure describing how to generate the values in the files appears below.

### 10.7.3.1 Creating a "Usershading" File

To create a "usershading" file and enable it, you must take the steps listed below. We strongly recommend that you read through all of the steps and read all of the other information in this section before you attempt to do shading correction.

The steps below are intended to give you the basic knowledge needed to create a "usershading" file and to activate it. **A code sample that includes the complete details of how to create a usershading file and how to enable shading correction on a camera is included with the Basler pylon SDK.**

When you create a "usershading" file you must make sure to create correction values for all of the pixels in the sensor's line regardless of how you plan to use the camera during normal operation.

## Creating a "Usershading" File for Offset Shading Correction

Creating a "usershading" file for offset shading correction will overwrite any "usershading" file for offset shading correction that is already in the camera's memory.

If you want to preserve the previous "usershading" file save it to your PC before creating the new "usershading" file.

For information about saving a "usershading" file to the PC, see Section 10.7.3.2 on page 153.

To create a "usershading" file for offset shading correction, perform the following steps:

1. Adjust the lighting, optics, line rate, exposure time control mode, exposure time, gain, and camera temperature as you would for normal operation.
2. Prevent light from striking the camera's sensor: Cover the camera lens, close the iris in the camera lens, or darken the room so that the camera will be capturing lines in complete darkness.
3. Set the camera's X Offset and Width parameters so that the entire width of the sensor will be used during frame acquisition.
   Note: You can create a "usershading" file for an AOI that is narrower than the entire width of the sensor. In this case, however, the "usershading" file will only apply to the narrower AOI or to smaller included AOIs. We recommend using the entire width of the sensor.
4. Select Offset Shading in the Basler pylon Viewer or via the Basler pylon API.
5. Select "usershading" file in the Basler pylon Viewer or via the Basler pylon API.
6. Go to the **Create** enumeration in the Basler pylon Viewer and select **Once** or send a create command via the Basler pylon API.
7. Perform at least 128 line acquisitions.

   To ease acquisition of the required number of lines we recommend to set the line start trigger mode to off for automatic line start triggering and to set the Height parameter for the frame to at least 128.

   For more information about the line start trigger mode, see Section 8.2.4 on page 77.

   For more information about defining a frame, see Section 8.1 on page 65.

   After 128 line acquisitions are completed the camera creates the "usershading" file automatically. The "usershading" file is stored in the camera's non-volatile memory and is not lost if the camera power is switched off.

Any time you make a change to the line rate, exposure time control mode, exposure time, gain, or camera temperature, you must create a new "usershading" file for offset shading correction. Using an out of date "usershading" file can result in poor image quality.

## Creating a "Usershading" File for Gain Shading Correction

Creating a "usershading" file for gain shading correction will overwrite any "usershading" file for gain shading correction that is already in the camera's memory.

If you want to preserve the previous "usershading" file save it to your PC before creating the new "usershading" file.

For information about saving a "usershading" file to the PC, see Section 10.7.3.2 on page 153.

To create a "usershading" file for gain shading correction, perform the following steps:

1. Adjust the lighting, optics, line rate, exposure time control mode, exposure time, gain, and camera temperature as you would for normal operation.
2. Place a uniform white target in the field of view of the camera.
3. Set the camera's X Offset and Width parameters so that the entire width of the sensor lines will be used during frame acquisition.
4. Perform several line acquisitions and examine the pixel values returned from the camera. The pixel values for the brightest pixels in each line should be about 90 to 95 % of maximum (i.e., if the camera is set for 8 bit output, the pixels should be from 90 to 95 % of 255).
   a. If the values for the brightest pixels are at 90 to 95 % of maximum, go on to step 5.
   b. If the values for the brightest pixels are not at 90 to 95 % of the maximum, adjust your lighting and/or lens aperture to achieve 90 to 95 %
5. Perform several line acquisitions and examine the pixel values returned from the camera. In each line, the values for the darkest pixels must be greater than 1/4 of the values for the brightest pixels. (If the values for the darkest pixels are less than 1/4 of the values for the brightest, the camera will not be able to fully correct for shading variations.)
   a. If the values for the darkest pixels are greater than 1/4 of the values for the brightest, go on to step 6.
   b. If the values for the darkest pixels are less than 1/4 of the values for the brightest pixels, it usually indicates extreme variations in lighting or poor quality optics. Make corrections as required.
6. Select Gain Shading in the Basler pylon Viewer or via the Basler pylon API.
7. Select "usershading" file in the Basler pylon Viewer or via the Basler pylon API.
8. Go to the **Create** enumeration in the Basler pylon Viewer and select **Once** or send a create command via the Basler pylon API.
9. Perform at least 128 line acquisitions.

   To ease acquisition of the required number of lines we recommend to set the line start trigger mode to off for automatic line start triggering and to set the Height parameter for the frame to at least 128.

   For more information about the line start trigger mode, see Section 8.2.4 on page 77.

   For more information about defining a frame, see Section 8.1 on page 65.

After 128 line acquisitions are completed the camera creates the "usershading" file automatically. The "usershading" file is stored in the camera's non-volatile memory and is not lost if the camera power is switched off.

Any time you make a change to the optics or lighting or if you change the camera's gain settings or exposure mode, you must create a new "usershading" file. Using an out of date "usershading" file can result in poor image quality.

### 10.7.3.2 Working with Shading Sets

Once you have created shading set files, you can use the following pylon API functions to work with the shading sets:

**Shading Selector** - is used to select the type of shading correction to configure, i.e. offset shading correction or gain shading correction.

**Shading Create** - is used to create a "usershading" file. The enumeration allows selecting the settings **Off** and **Once**.

**Shading Enable** - is used to enable and disable the selected type of shading correction.

**Shading Set Selector** - is used to select the shading set to which the activate and the create enumeration commands will be applied.

**Shading Set Activate** - is used to activate the selected shading set. "Activate" means that the shading set will be copied from the camera's non-volatile memory into it's volatile memory. When the shading correction feature is enabled, the shading set in the volatile memory will be used to perform shading correction.

**Shading Set Default Selector** - is used to select the shading set that will be loaded into the camera's volatile memory during camera bootup.

**Shading Status** - is used to determine the error status of operations such as Shading Set Activate. The following error statuses may be indicated:

No error - the last operation performed was successful.

Startup Set error - there was a problem with the default shading set.

Activate error - the selected shading set could not be loaded into the volatile memory.

Create error - and error occurred during the attempt of creating a "usershading" file.

The use of the pylon API functions listed above is illustrated in the shading correction sample code included with the pylon SDK.

You can also use the Shading parameters group in the Basler pylon Viewer application to access these functions.

And you can use the File Access selection in the Camera menu of the Viewer to save a shading set file to a PC and to upload a shading set file from the PC to the camera.

# 10.8 Trigger Delay

The trigger delay feature lets you specify a delay that will be applied between the receipt of a hardware acquisition start trigger or frame start trigger and it becoming effective.

The trigger delay may be specified as a time interval in the range from 0 to 1000000 µs (equivalent to 1 s) or as a number of consecutive line start triggers where the maximum number depends on the camera model. When the delay is set to 0 µs or 0 line start triggers, no delay will be applied.

The trigger delay will not operate when the camera is triggered by your application software and when the camera operates in continuous frame mode (free run).

When setting the trigger delay you must specify the kind of trigger to be delayed (acquisition start or frame start trigger) and the extend of the delay expressed as a time interval or as a number of consecutive line start triggers.

You can set the trigger delay from within your application software by using the pylon API. As examples, the following code snippets illustrate using the API to set the delay for the acquisition start trigger to 1000 µs and to set the delay for the frame start trigger to 100 line start triggers:

```
// Trigger delay
Camera.TriggerSource = AcquisitionStart;
double TriggerDelay_us = 1000.0   // 1000us == 1ms == 0.001s;
Camera.TriggerDelaySource = TriggerDelay_us;
Camera.TriggerDelayAbs.SetValue( TriggerDelay_us );
```

```
// Trigger delay
Camera.TriggerSource = FrameStart;
int NumberLineTriggers = 100;
Camera.TriggerDelaySource = LineTrigger;
Camera.TriggerDelayLineTriggerCount.SetValue( NumberLineTriggers );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

# 10.9 Error Codes

The camera can detect several user correctable errors. If one of these errors is present, the camera will set an error code and will flash both the yellow and green LEDs in the LED indicator.

The following table indicates the available error codes:

| Code | Condition | Meaning |
|---|---|---|
| 0 | No Error | The camera has not detected any errors since the last time that the error memory was cleared. |
| 1 | Overtrigger | An overtrigger has occurred.<br>The user has applied an acquisition start trigger to the camera when the camera was not in a waiting for acquisition start condition.<br>Or, the user has applied a frame start trigger to the camera when the camera was not in a waiting for frame start condition. |
| 2 | User set load | An error occurred when attempting to load a user set.<br>Typically, this means that the user set contains an invalid value. Try loading a different user set. |
| 3 | Invalid Parameter | A parameter is set out of range or in an otherwise invalid manner. |

Table 10: Error Codes

When the camera detects a user correctable error, it sets the appropriate error code in an error memory. If two or three different detectable errors have occurred, the camera will store the code for each type of error that it has detected (it will store one occurrence of the each code no matter how many times it has detected the corresponding error).

You can use the following procedure to check the error codes:

- Read the value of the Last Error parameter. The Last Error parameter will indicate the last error code stored in the memory.
- Execute the Clear Last Error Command to clear the last error code from the memory.
- Continue reading and clearing the last error until the parameter indicates a No Error code.

### Reading and Clearing the Error Codes Using Basler Pylon

You can use the pylon API to read the value of the Last Error parameter and to execute a Clear Last Error command from within your application software. The following code snippets illustrate using the API to read the parameter value and execute the command:

```
// Read the value of the last error code in the memory
LastErrorEnums lasterror = Camera.LastError.GetValue();

// Clear the value of the last error code in the memory
Camera.ClearLastError.Execute( );
```

You can also use the Basler pylon Viewer application to easily set the parameter and execute the command.

# 10.10Test Images

All cameras include the ability to generate test images. Test images are used to check the camera's basic functionality and its ability to transmit an image to the host PC. Test images can be used for service purposes and for failure diagnostics.

When the camera is in test image mode, the optics, imaging sensor, and the ADCs are not used. The lines that make up each test image are generated internally by the camera's logic and the generated lines are collected in frame memory. When each test image frame is complete, it will be transmitted to the host PC in the same manner as with normal camera operation. The size of each test image frame will be determined by the frame parameter settings as with normal operation.

If the camera is set to use an electrical signal applied to input line 1, line 2, or line 3 as the source signal for the frame trigger and/or the line trigger, these signals must be provided to the camera in order to generate test images.

## The Effect of Camera Settings on Test Images

When any test image is active, the camera's analog features such as analog gain, black level, and exposure time have no effect on the images transmitted by the camera. For test images 1, 2, and 3, the camera's digital features, will also have no effect on the transmitted images. But for test images 4 and 5, the cameras digital features will affect the images transmitted by the camera.

## Enabling a Test Image

The Test Image Selector is used to set the camera to output a test image. You can set the value of the Test Image Selector to one of the test images or to "test image off".

You can set the Test Image Selector from within your application software by using the pylon API. The following code snippets illustrate using the API to set the selector:

```
// set for no test image
Camera.TestImageSelector.SetValue( TestImageSelector_Off );

// set for the first test image
Camera.TestImageSelector.SetValue( TestImageSelector_Testimage1 );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

# 10.10.1Test Images in Detail

### Test Image 1 - Fixed Diagonal Gray Gradient (8 bit)

The 8 bit fixed diagonal gray gradient test image is best suited for use when the camera is set for monochrome 8 bit output. The test image consists of fixed diagonal gray gradients ranging from 0 to 255.

If the camera is set for 8 bit output, test image one will look similar to Fig. 42.

The mathematical expression for this test image is:

Gray Value = [column number + row number] MOD 256

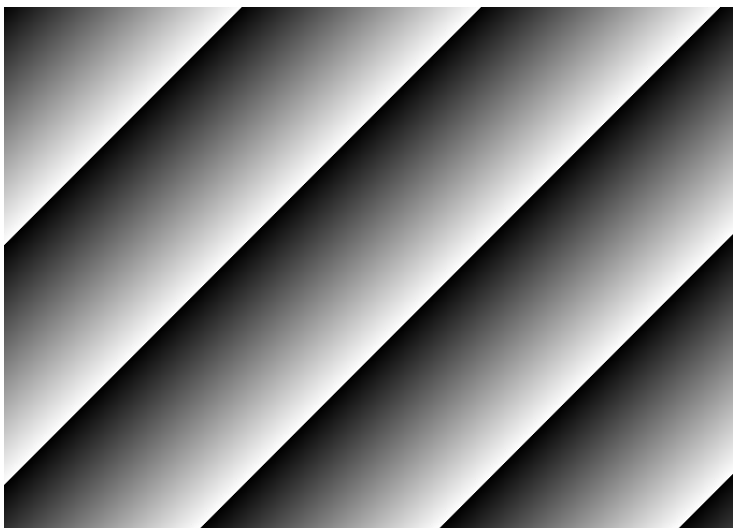


Fig. 42: Test Image One

### Test Image 2 - Moving Diagonal Gray Gradient (8 bit)

The 8 bit moving diagonal gray gradient test image is similar to test image 1, but it is not stationary. The image moves by one pixel from right to left whenever a new frame acquisition is initiated. The test pattern uses a counter that increments by one for each new frame acquisition.

The mathematical expression for this test image is:

Gray Value = [column number + row number + counter] MOD 256

### Test Image 3 - Moving Diagonal Gray Gradient (12 bit)

The 12 bit moving diagonal gray gradient test image is similar to test image 2, but it is a 12 bit pattern. The image moves by one pixel from right to left whenever a new frame acquisition is initiated. The test pattern uses a counter that increments by one for each new frame acquisition.

The mathematical expression for this test image is:

Gray Value = [column number + row number + counter] MOD 4096

### Test Image 4 - Moving Diagonal Gray Gradient Feature Test (8 bit)

The basic appearance of test image 4 is similar to test image 2 (the 8 bit moving diagonal gray gradient image). The difference between test image 4 and test image 2 is this: if a camera feature that involves digital processing is enabled, test image 4 **will** show the effects of the feature while test image 2 **will not**. This makes test image 4 useful for checking the effects of digital features such as spatial correction.

### Test Image 5 - Moving Diagonal Gray Gradient Feature Test (12 bit)

The basic appearance of test image 5 is similar to test image 3 (the 12 bit moving diagonal gray gradient image). The difference between test image 5 and test image 3 is this: if a camera feature that involves digital processing is enabled, test image 5 **will** show the effects of the feature while test image 3 **will not**. This makes test image 5 useful for checking the effects of digital features.

# 10.11 Device Information Parameters

Each camera includes a set of "device information" parameters. These parameters provide some basic information about the camera. The device information parameters include:

- Device Vendor Name (read only) - contains the name of the camera's vendor. This string will always indicate Basler as the vendor.
- Device Model Name (read only) - contains the model name of the camera, for example, raL2048-48gm.
- Device Manufacturer Info (read only) - can contain some information about the camera manufacturer. This string usually indicates "none".
- Device Version (read only) - contains the device version number for the camera.
- Firmware Version (read only) - contains the version of the firmware in the camera.
- Device ID (read only) - contains the serial number of the camera.
- Device User ID (read / write) - is used to assign a user defined name to a device. This name will be displayed in the Basler pylon Viewer and the Basler pylon IP Configuration Tool. The name will also be visible in the "friendly name" field of the device information objects returned by pylon's device enumeration procedure.
- Device Scan Type (read only) - contains the scan type of the camera, for example, line scan.
- Sensor Width (read only) - contains the physical width of the sensor in pixels.
- Sensor Height (read only) - contains the physical height of the sensor in pixels.
- Max Width (read only) - Indicates the camera's maximum width setting.
- Max Height (read only) - Indicates the camera's maximum height setting.
- Sensor Board Temperature (read only) - Indicates the current temperature in degrees centigrade of the camera's sensor board.

You can read the values for all of the device information parameters or set the value of the Device User ID parameter from within your application software by using the pylon API. The following code snippets illustrate using the API to read the parameters or write the Device User ID:

```
// Read the Vendor Name parameter
Pylon::String_t vendorName = Camera.DeviceVendorName.GetValue();

// Read the Model Name parameter
Pylon::String_t modelName = Camera.DeviceModelName.GetValue();

// Read the Manufacturer Info parameter
Pylon::String_t manufacturerInfo = Camera.DeviceManufacturerInfo.GetValue();

// Read the Device Version parameter
Pylon::String_t deviceVersion = Camera.DeviceVersion.GetValue();

// Read the Firmware Version parameter
Pylon::String_t firmwareVersion = Camera.DeviceFirmwareVersion.GetValue();
```

```
// Read the Device ID parameter
Pylon::String_t deviceID = Camera.DeviceID.GetValue();

// Write and read the Device User ID
Camera.DeviceUserID = "custom name";
Pylon::String_t deviceUserID = Camera.DeviceUserID.GetValue();

// Read the Sensor Width parameter
int64_t sensorWidth = Camera.SensorWidth.GetValue();

// Read the Sensor Height parameter
int64_t sensorHeight = Camera.SensorHeight.GetValue();

// Read the Max Width parameter
int64_t maxWidth = Camera.WidthMax.GetValue();

// Read the Max Height parameter
int64_t maxHeight = Camera.HeightMax.GetValue();

// Read the Temperature Abs parameter
camera.TemperatureSelector.SetValue(TemperatureSelector_Sensorboard);
double temperature = camera.TemperatureAbs.GetValue();
```

You can also use the Basler pylon Viewer application to easily read the parameters and to read or write the Device User ID.

You can use the Basler pylon IP Configuration tool to read or write the Device User ID.

For more information about the pylon Viewer, see Section 3.1 on page 17.

For detailed information about using the pylon API and the pylon IP Configuration Tool, refer to the Basler pylon Programmer’s Guide and API Reference.

# 10.12 User Defined Values

The camera can store two "user defined values". These two values are 32 bit signed integer values that you can set and read as desired. They simply serve as convenient storage locations for the camera user and have no impact on the operation of the camera.

The two values are designated as Value 1 and Value 2.

## Setting User Defined Values

Setting a user defined value using Basler pylon is a two step process:

- Set the User Defined Value Selector to Value 1 or Value 2.
- Set the User Defined Value parameter to the desired value for the selected value.

You can use the pylon API to set the User Defined Value Selector and the User Defined Value parameter value from within your application software. The following code snippet illustrates using the API to set the selector and the parameter value:

```
// Set user defined value 1
Camera.UserDefinedValueSelector.SetValue( UserDefinedValueSelector_Value1 );
Camera.UserDefinedValue.SetValue( 1000 );

// Set user defined value 2
Camera.UserDefinedValueSelector.SetValue( UserDefinedValueSelector_Value2 );
Camera.UserDefinedValue.SetValue( 2000 );

// Get the value of user defined value 1
Camera.UserDefinedValueSelector.SetValue( UserDefinedValueSelector_Value1 );
int64_t UserValue1 = Camera.UserDefinedValue.GetValue();
```

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the Basler pylon API and the pylon Viewer, see Section 3.1 on page 17.

# 10.13 Configuration Sets

A configuration set is a group of values that contains all of the parameter settings needed to control the camera. There are three basic types of configuration sets: the active configuration set, the default configuration set, and user configuration sets.

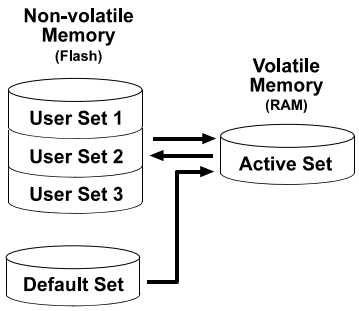



Fig. 43: Configuration Sets

## Active Configuration Set

The active configuration set contains the camera's current parameter settings and thus determines the camera's performance, that is, what your image currently looks like. When you change parameter settings using the pylon API or the pylon Viewer, you are making changes to the active configuration set. The active configuration set is located in the camera's volatile memory and the settings are lost if the camera is reset or if power is switched off. The active configuration set is usually called the "active set" for short.

## Default Configuration Set

When a camera is manufactured, a test setup is performed on the camera and an optimized configuration is determined. The default configuration set contains the camera's factory optimized configuration. The default configuration set is saved in a permanent file in the camera's non-volatile memory. It is not lost when the camera is reset or switched off and it cannot be changed. The default configuration set is usually just called the "default set" for short.

## User Configuration Sets

As mentioned above, the active configuration set is stored in the camera's volatile memory and the settings are lost if the camera is reset or if power is switched off. The camera can save most of the settings from the current active set to a reserved area in the camera's non-volatile memory. A configuration set saved in the non-volatile memory is not lost when the camera is reset or switched off. There are three reserved areas in the camera's non-volatile memory available for saving configuration sets. A configuration set saved in a reserved area is commonly referred to as a "user configuration set" or "user set" for short.

The three available user sets are called User Set 1, User Set 2, and User Set 3.

| | |
|---|---|
| ⓘ | The settings for frame transmission delay, inter packet delay, and the luminance lookup table are not saved in the user sets and are lost when the camera is reset or switched off. If used, these settings must be set again after each camera reset or restart. |

### Default Startup Set

You can select the default configuration set or one of the user configuration sets stored in the camera's non-volatile memory to be the "default startup set." The configuration set that you designate as the default startup set will be loaded into the active set whenever the camera starts up at power on or after a reset. Instructions for selecting the default startup set appear on the next page.

## 10.13.1 Saving Configuration Sets

Saving the current active set into a user set in the camera's non-volatile memory is a three step process:

- Make changes to the camera's settings until the camera is operating in a manner that you would like to save.
- Set the User Set Selector to User Set 1, User Set 2, or User Set 3.
- Execute a User Set Save command to save the active set to the selected user set.

Saving an active set to a user set in the camera's non-volatile memory will overwrite any parameters that were previously saved in that user set.

You can set the User Set Selector and execute the User Set Save command from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and execute the command:

```
Camera.UserSetSelector.SetValue( UserSetSelector_UserSet1 );
Camera.UserSetSave.Execute( );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

## 10.13.2 Loading a Saved Set or the Default Set into the Active Set

If you have saved a configuration set into the camera's non-volatile memory, you can load the saved set from the camera's non-volatile memory into the camera's active set. When you do this, the loaded set overwrites the parameters in the active set. Since the settings in the active set control the current operation of the camera, the settings from the loaded set will now be controlling the camera.

You can also load the default set into the camera's active set.

To load a saved configuration set or the default set from the camera's non-volatile memory into the active set:

- Set the User Set Selector to User Set 1, User Set 2, User Set 3, or Default.
- Execute a User Set Load command to load the selected set into the active set.

You can set the User Set Selector and execute the User Set Load command from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector and execute the command:

```
Camera.UserSetSelector.SetValue( UserSetSelector_UserSet2 );
Camera.UserSetLoad.Execute( );
```

Loading a user set or the default set into the active set is only allowed when the camera is idle, i.e. when it is not acquiring lines.

Loading the default set into the active set is a good course of action if you have grossly misadjusted the settings in the camera and you are not sure how to recover. The default settings are optimized for use in typical situations and will provide good camera performance in most cases.

## 10.13.3 Selecting the Default Startup Set

You can select the default configuration set or one of the user configuration sets stored in the camera's non-volatile memory to be the "default startup set". The configuration set that you designate as the default startup set will be loaded into the active set whenever the camera starts up at power on or after a reset.

The User Set Default Selector is used to select the default startup set:

- Set the User Set Default Selector to User Set 1, User Set 2, User Set 3, or Default.

You can set the User Set Default Selector from within your application software by using the pylon API. The following code snippet illustrates using the API to set the selector:

```
Camera.UserSetDefaultSelector.SetValue( UserSetDefaultSelector_Default );
```

# 11 Chunk Features

This section provides detailed information about the chunk features available on each camera.

## 11.1 What are Chunk Features?

In most cases, enabling a camera feature will simply change the behavior of the camera. The Test Image feature is a good example of this type of camera feature. When the Test Image feature is enabled, the camera outputs a test image rather than an acquired image. This type of feature is referred to as a "standard" feature.

When certain camera features are enabled, the camera actually develops some sort of information about each frame that it acquires. In these cases, the information is added to each frame as a trailing data "chunk" when the image is transferred to the host PC. Examples of this type of camera feature are the Frame Counter feature and the Time Stamp feature. When the Frame Counter feature is enabled, for example, after a frame is acquired, the camera checks a counter that tracks the number of frames acquired and develops a frame counter stamp for the frame. And if the Time Stamp feature is enabled, the camera creates a time stamp indicating when the frame was acquired. The frame counter stamp and the time stamp would be added as "chunks" of trailing data to each frame as the frame is transmitted from the camera. The features that add chunks to the acquired frames are referred to as “chunk” features.

Before you can use any of the features that add chunks to the frames, you must make the chunk mode active. Making the chunk mode active is described in the next section.

# 11.2 Making the "Chunk Mode" Active and Enabling the Extended Data Stamp

Before you can use any of the camera's "chunk" features, the "chunk mode" must be made active. Making the chunk mode active does two things:

- It automatically enables the Extended Frame Data chunk feature.
- It makes the camera's other chunk features available to be enabled.

To make the chunk mode active:

- Set the Chunk Mode Active parameter to true.

You can set the Chunk Mode Active parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to set the parameter value:

```
Camera.ChunkModeActive.SetValue( true );
```

Note that making the chunk mode inactive switches all chunk features off.

Also note that when you enable ChunkModeActive, the PayloadType for the camera changes from "Pylon::PayloadType_Image" to "Pylon::PayloadType_ChunkData".

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

When the chunk mode made is active, the Extended Frame Data feature will automatically be enabled, and the camera will add an "extended frame data" chunk to each acquired image. The extended frame data chunk appended to each acquired image contains some basic information about the frame. The information contained in the chunk includes:

- The X Offset, Width, and Height settings for the frame
- The Pixel Format of the image data in the frame
- The Minimum Dynamic Range and the Maximum Dynamic Range

To retrieve data from the extended frame data chunk appended to a frame that has been received by your PC, you must first run the frame and its appended chunks through the chunk parser included in the pylon API. Once the chunk parser has been used, you can retrieve the extended frame data by doing the following:

- Read the value of the Chunk Offset X parameter.
- Read the value of the Chunk Width parameter.
- Read the value of the Chunk Height parameter.
- Read the value of the Chunk Pixel Format parameter.
- Read the value of the Chunk Dynamic Range Min.
- Read the value of the Chunk Dynamic Range Max.

The following code snippet illustrates using the pylon API to run the parser and retrieve the extended image data:

```
// retrieve data from the extended frame data chunk
IChunkParser &ChunkParser = *Camera.CreateChunkParser();
GrabResult Result;
StreamGrabber.RetrieveResult( Result );
ChunkParser.AttachBuffer( (unsigned char*) Result.Buffer(),
  Result.GetPayloadSize() );
int64_t offsetX = Camera.ChunkOffsetX.GetValue();
int64_t width = Camera.ChunkWidth.GetValue();
int64_t height = Camera.ChunkHeight.GetValue();
int64_t dynamicRangeMin = Camera.ChunkDynamicRangeMin.GetValue();
int64_t dynamicRangeMax = Camera.ChunkDynamicRangeMax.GetValue();
ChunkPixelFormatEnums pixelFormat = Camera.ChunkPixelFormat.GetValue();
```

For more information about using the chunk parser, see the sample code that is included with the Basler pylon Software Development Kit (SDK).

For more information about the pylon Viewer, see Section 3.1 on page 17.

# 11.3 Frame Counter

The Frame Counter feature numbers frames sequentially as they are acquired. When the feature is enabled, a chunk is added to each completed frame containing the value of the counter.

The frame counter is a 32 bit value. The counter starts at 0 and wraps back to 0 after it reaches its maximum. The counter increments by 1 for each acquired frame. Whenever the camera is powered off, the counter will reset to 0.

Be aware that if the camera is acquiring frames continuously and continuous acquisition is stopped, several numbers in the counting sequence may be skipped. This happens due to the internal buffering scheme used in the camera.

The chunk mode must be made active before you can enable the frame counter feature or any of the other chunk features. Making the chunk mode inactive disables all chunk features.

### Enabling the Frame Counter and Retrieving Chunk Data

To enable the Frame Counter chunk:

- Use the Chunk Selector to select the Frame Counter chunk.
- Use the Chunk Enable parameter to set the value of the chunk to true.

Once the frame counter chunk is enabled, the camera will add a frame counter chunk to each acquired frame.

To retrieve data from a chunk appended a frame that has been received by your PC, you must first run the frame and its appended chunks through the chunk parser included in the pylon API. Once the chunk parser has been used, you can retrieve the frame counter information by doing the following:

- Read the value of the Chunk Frame Counter parameter.

You can set the Chunk Selector and the Chunk Enable parameter value from within your application software by using the pylon API. You can also run the parser and retrieve the chunk data. The following code snippets illustrate using the API to activate the chunk mode, enable the frame counter chunk, run the parser, and retrieve the frame counter chunk data:

```
// make chunk mode active and enable Frame Counter chunk
Camera.ChunkModeActive.SetValue( true );
Camera.ChunkSelector.SetValue( ChunkSelector_Framecounter );
Camera.ChunkEnable.SetValue( true );

// retrieve date from the chunk
IChunkParser &ChunkParser = *Camera.CreateChunkParser();
GrabResult Result;
StreamGrabber.RetrieveResult( Result );
ChunkParser.AttachBuffer( (unsigned char*) Result.Buffer(),
```

```
    Result.GetPayloadSize() );
int64_t frameCounter = Camera.ChunkFramecounter.GetValue();
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

## Frame Counter Reset

Whenever the camera is powered off, the frame counter will reset to 0. During operation, you can reset the frame counter via I/O input 1, I/O input 2, I/O input 3 or software, and you can disable the reset. By default, the frame counter reset is disabled.

To use the frame counter reset:

- Configure the frame counter reset by setting the counter selector to Counter2 and setting the counter event source to FrameStart.
- Set the counter reset source to Line1, Line2, Line3, Software or to Off.
- Execute the command if using software as the counter reset source.

You can set the frame counter reset parameter values from within your application software by using the pylon API. The following code snippets illustrate using the API to configure and set the frame counter reset and to execute a reset via software.

```
// configure reset of frame counter
Camera.CounterSelector.SetValue( CounterSelector_Counter2 );
Camera.CounterEventSource.SetValue( CounterEventSource_FrameStart );

// select reset by signal on input line 1
Camera.CounterResetSource.SetValue( CounterResetSource_Line1 );

// select reset by signal on input line 2
Camera.CounterResetSource.SetValue( CounterResetSource_Line2 );

// select reset by signal on input line 3
Camera.CounterResetSource.SetValue( CounterResetSource_Line3 );

// select reset by software
Camera.CounterResetSource.SetValue( CounterResetSource_Software );
// execute reset by software
Camera.CounterReset.Execute();

// disable reset
Camera.CounterResetSource.SetValue( CounterResetSource_Off );
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

# 11.4 Time Stamp

The Time Stamp feature adds a chunk to each acquired frame. The chunk contains a time stamp that was generated when the frame start trigger for the frame became valid.

Note that when the camera is set for continuous acquisition mode with the frame start trigger set to off, the user is not required to apply frame start trigger signals to the camera. In this case, the camera will internally generate a signal that will be used for the stamp.

The time stamp is a 64 bit value. The time stamp is based on a counter that counts the number of "time stamp clock ticks" generated by the camera. The unit for each tick is 8 ns (as specified by the Gev Timestamp Tick Frequency). The counter starts at camera reset or at power off/on.

The chunk mode must be made active before you can enable the time stamp feature or any of the other chunk features. Making the chunk mode inactive disables all chunk features.

### Enabling the Time Stamp and Retrieving Chunk Data

To enable the Time Stamp chunk:

- Use the Chunk Selector to select the Time Stamp chunk.
- Use the Chunk Enable parameter to set the value of the chunk to true.

Once the time stamp chunk is enabled, the camera will add a time stamp chunk to each acquired frame.

To retrieve data from a chunk appended to a frame that has been received by your PC, you must first run the frame and its appended chunks through the chunk parser that is included in the pylon API. Once the chunk parser has been used, you can retrieve the time stamp information by doing the following:

- Read the value of the Chunk Time Stamp parameter.

You can set the Chunk Selector and the Chunk Enable parameter value from within your application software by using the pylon API. You can also run the parser and retrieve the chunk data. The following code snippets illustrate using the API to activate the chunk mode, enable the time stamp chunk, run the parser, and retrieve the frame counter chunk data:

```
// make chunk mode active and enable Time Stamp chunk
Camera.ChunkModeActive.SetValue( true );
Camera.ChunkSelector.SetValue( ChunkSelector_Timestamp );
Camera.ChunkEnable.SetValue( true );

// retrieve data from the chunk
IChunkParser &ChunkParser = *Camera.CreateChunkParser();
GrabResult Result;
StreamGrabber.RetrieveResult( Result );
ChunkParser.AttachBuffer( (unsigned char*) Result.Buffer(),
```

```
    Result.GetPayloadSize() );
int64_t timeStamp = Camera.ChunkTimestamp.GetValue();
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

# 11.5 Trigger Counters

The camera has the following "trigger counters" available that can help you determine if you are triggering the camera correctly:

- the Line Trigger Ignored Counter
- the Frame Trigger Ignored Counter
- the Line Trigger End To End Counter
- the Frame Trigger Counter
- the Frames Per Trigger Counter

When a counter is enabled, a chunk is added to each completed frame containing the value of the counter. So if you have all five counters enabled, for example, five chunks will be added to each frame.

The chunk mode must be made active before you can enable the time stamp feature or any of the other chunk features. Making the chunk mode inactive disables all chunk features.

The Line Trigger Ignored, Frame Trigger Ignored, and Line Trigger End To End counters are each 32 bit counters. The Frame Trigger and Frames Per Trigger counters are each 16 bit counters.

### Line Trigger Ignored Counter

The Line Trigger Ignored Counter counts the number of line triggers that were received during the acquisition of the current frame but were ignored (not acted on). A line trigger will be ignored if the camera is already in the process of acquiring a line when the trigger is received. Typically, this will happen if you "overtrigger" the camera, i.e., try to acquire lines at a rate that is higher than allowed. The magnitude of this counter will give you an idea of how badly the camera is being overtriggered. The higher the counter, the worse the overtriggering.

### Frame Trigger Ignored Counter

The Frame Trigger Ignored Counter counts the number of frame triggers that were not acted upon during the acquisition of the frame because the camera was not ready to accept the trigger. Typically, this will happen if you attempt to trigger the start of a new frame while the camera is currently in the process of acquiring a frame.

## Line Trigger End To End Counter

The Line Trigger End to End Counter counts the number of line triggers received by the camera from the end of the previous frame acquisition to the end of the current frame acquisition. If you subtract the number of lines actually included in the current frame from the number of lines shown by this counter, it will tell you the number of line triggers that were received but not acted on during the frame end to frame end period.

## Frame Trigger Counter and Frames Per Trigger Counter

The Frame Trigger Counter and the Frames Per Trigger Counter are designed to be used together. They are available when the frame start trigger activation is set to either Level High or Level Low. The Frame Trigger Counter counts the number of frame trigger valid periods, and it increments each time the frame trigger becomes valid. The Frames Per Trigger counter counts the number of frames acquired during each frame valid period. The counter increments for each acquired frame (also for partial frames) and resets to zero for each new frame valid period. The way that the counters work is illustrated below.

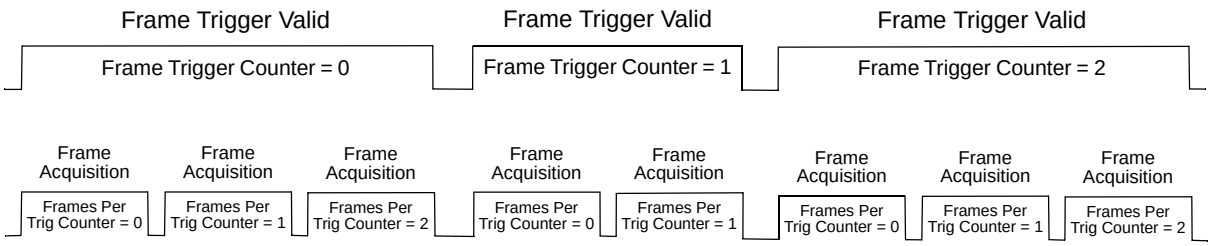

Frame Trigger Valid
Frame Trigger Valid
Frame Trigger Valid
Frame Trigger Counter = 0
Frame Trigger Counter = 1
Frame Trigger Counter = 2
Frame Acquisition
Frames Per Trig Counter = 0
Frame Acquisition
Frames Per Trig Counter = 1
Frame Acquisition
Frames Per Trig Counter = 2
Frame Acquisition
Frames Per Trig Counter = 0
Frame Acquisition
Frames Per Trig Counter = 1
Frame Acquisition
Frames Per Trig Counter = 0
Frame Acquisition
Frames Per Trig Counter = 1
Frame Acquisition
Frames Per Trig Counter = 2

Fig. 44: Frame Trigger Counter and Frames Per Trigger Counter

These two counters can be used to determine which frames were acquired during a particular frame trigger valid period. This information will be especially useful in a situation where several frames must be stitched together to form an image of a single large object.

## Enabling the Trigger Counters and Retrieving Chunk Data

To enable the one of the trigger counter chunks:

- Use the Chunk Selector to select the chunk.
- Use the Chunk Enable parameter to set the value of the chunk to true.

Once a trigger counter chunk has been enabled, the camera will add the counter chunk to each acquired frame.

You can set the Chunk Selector and Chunk Enable parameter value from within your application software by using the pylon API. The following code snippet illustrates using the API to activate the chunk mode and enable the trigger counter chunks:

```
// make chunk mode active
Camera.ChunkModeActive.SetValue( true );
```

```
// enable the trigger counter chunks
Camera.ChunkSelector.SetValue( ChunkSelector_LineTriggerIgnoredCounter );
Camera.ChunkEnable.SetValue( true );
Camera.ChunkSelector.SetValue( ChunkSelector_FrameTriggerIgnoredCounter );
Camera.ChunkEnable.SetValue( true );
Camera.ChunkSelector.SetValue( ChunkSelector_LineTriggerEndToEndCounter );
Camera.ChunkEnable.SetValue( true );
Camera.ChunkSelector.SetValue( ChunkSelector_FrameTriggerCounter );
Camera.ChunkEnable.SetValue( true );
Camera.ChunkSelector.SetValue( ChunkSelector_FramesPerTriggerCounter );
Camera.ChunkEnable.SetValue( true );
```

To retrieve data from a chunk appended a frame that has been received by your PC, you must first run the frame and its appended chunks through the chunk parser included in the pylon API. Once the chunk parser has been used, you can retrieve the counter values from the chunks by doing the following:

- Read the value of the Chunk Line Trigger Ignored Counter parameter.
- Read the value of the Chunk Frame Trigger Ignored Counter parameter.
- Read the value of the Chunk Line Trigger End To End Counter parameter.
- Read the value of the Chunk Frame Trigger Counter parameter.
- Read the value of the Chunk Frames Per Trigger Counter parameter.

You can run the chunk parser and retrieve the counter values from within your application software by using the pylon API. The following code snippet illustrates using the API to run the parser and retrieve the frame counter chunk data:

```
// run the chunk parser
IChunkParser &ChunkParser = *Camera.CreateChunkParser();
GrabResult Result;
StreamGrabber.RetrieveResult( Result );
ChunkParser.AttachBuffer( (unsigned char*) Result.Buffer(),
   Result.GetPayloadSize() );

// retrieve data from the chunks
int64_t LTIgnoredCounter = Camera.ChunkLineTriggerIgnoredCounter.GetValue();
int64_t FTIgnoredCounter = Camera.ChunkFrameTriggerIgnoredCounter.GetValue();
int64_t LTEECounter = Camera.ChunkLineTriggerEndToEndCounter.GetValue();
int64_t FTCounter = Camera.ChunkFrameTriggerCounter.GetValue();
int64_t FPTCounter = Camera.ChunkFramesPerTriggerCounter.GetValue();
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3.1 on page 17.

# 11.6 Encoder Counter

The encoder counter chunk indicates the value of the Shaft Encoder Module Counter parameter at the time of the occurrence of a frame trigger. When the encoder counter chunk is enabled, a chunk is added to each frame containing the value of the Shaft Encoder Module Counter parameter. The encoder counter chunk is a 16 bit value. The minimum value is 0 and the maximum is 32767.

The Shaft Encoder Module Counter is part of the shaft encoder module. See the Shaft Encoder Module section for more information amount the shaft encoder module, about the Shaft Encoder Module Counter, and about its possible modes of incrementing.

The chunk mode must be made active before you can enable the time stamp feature or any of the other chunk features. Making the chunk mode inactive disables all chunk features.

To enable the encoder counter chunk:

- Use the Chunk Selector to select the Encoder Counter chunk.
- Use the Chunk Enable parameter to set the value of the chunk to true.

Once the encoder counter chunk is enabled, the camera will add an encoder counter chunk to each acquired frame.

To retrieve data from a chunk appended to an frame that has been received by your PC, you must first run the frame and its appended chunks through the chunk parser included in the pylon API. Once the chunk parser has been used, you can retrieve the encoder counter information by doing the following:

- Read the value of the Chunk Encoder Counter parameter.

You can set the Chunk Selector and Chunk Enable parameter value from within your application software by using the pylon API. You can also run the parser and retrieve the chunk data. The following code snippets illustrate using the API to activate the chunk mode, enable the encoder counter chunk, run the parser, and retrieve the encoder counter chunk data:

```
// make chunk mode active and enable Encoder Counter chunk
Camera.ChunkModeActive.SetValue( true );
Camera.ChunkSelector.SetValue( ChunkSelector_ChunkShaftEncoderCounter );
Camera.ChunkEnable.SetValue( true );

// retrieve date from the chunk
IChunkParser &ChunkParser = *Camera.CreateChunkParser();
GrabResult Result;
StreamGrabber.RetrieveResult( Result );
ChunkParser.AttachBuffer( (unsigned char*) Result.Buffer(),
   Result.GetPayloadSize() );
int64_t EncoderCounter = Camera.ChunkShaftEncoderCounter.GetValue();
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

# 11.7 Input Line Status At Line Trigger

The Input Status At Line Trigger feature samples the status of all of the camera's input lines each time a line acquisition is triggered. It collects the input line status data for each acquired line in a chunk and adds the chunk to the frame that includes the acquired line.

The input status at line trigger information is a 4 bit value. As shown in Fig. 45, certain bits in the value are associated with each line and the bits will indicate the state of the lines. If a bit is 0, it indicates that the state of the associated line was low at the time of triggering. If a bit is 1, it indicates that the state of the associated line was high at the time of triggering.

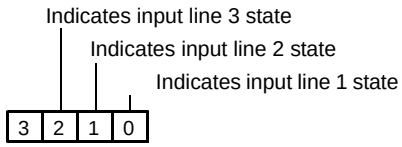



Fig. 45: Input Status At Line Trigger Parameter Bits

**Note**

The chunk mode must be active before you can enable the input status at line trigger chunk or any of the other chunk feature. Making the chunk mode inactive disables all chunk features.

The maximum for the Height parameter value is 1024 if the input status at line trigger chunk is enabled. Other conditions may further decrease the maximum parameter value. For more information, see the Defining a Frame section.

The chunk mode must be made active before you can enable the time stamp feature or any of the other chunk features. Making the chunk mode inactive disables all chunk features.

The maximum for the Height parameter value is 1024 if the input status at line trigger chunk is enabled. Other conditions may further decrease the maximum parameter value. For more information, see the Defining a Frame section.

To enable the input status at line trigger chunk:

- Use the Chunk Selector to select the Input Status At Line Trigger chunk.
- Use the Chunk Enable parameter to set the value of the chunk to true.

Once the input status at line trigger chunk is enabled, the camera will add an input status at line trigger chunk to each acquired frame.

To retrieve data from a chunk appended to an image that has been received by your PC, you must first run the image and its appended chunks through the chunk parser included in the pylon API. Once the chunk parser has been used, you can retrieve the input line status at line trigger information that was extant when acquisition of line i was triggered by doing the following:

- Read the value of the Chunk Input Status At Line Trigger parameter.

You can set the Chunk Selector and Chunk Enable parameter value from within your application software by using the pylon API. You can also run the parser and retrieve the chunk data. The following code snippets illustrate using the API to activate the chunk mode, enable the input status at line trigger chunk, run the parser, and retrieve the input status at line trigger chunk data for the acquired line i:

```
Camera.ChunkModeActive.SetValue(true);
Camera.ChunkSelector.SetValue(ChunkSelector_InputStatusAtLineTrigger);
Camera.ChunkEnable.SetValue(true);

// grab image and feed it to the chunk parser ...

int MaxIdx = int(Camera.ChunkInputStatusAtLineTriggerIndex.GetMax());
for (int i = 0; i <= MaxIdx; i++)
{
    Camera.ChunkInputStatusAtLineTriggerIndex.SetValue(i);
    int value = int(Camera.ChunkInputStatusAtLineTriggerValue.GetValue());
    printf("State of inputs at line %d: %X\n", i, value);
   }
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer’s Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

# 11.8 CRC Checksum

The CRC (Cyclic Redundancy Check) Checksum feature adds a chunk to each acquired frame containing a CRC checksum calculated using the X-modem method. As shown in Fig. 46 on page 182, the checksum is calculated using all of the image data in the frame and all of the appended chunks except for the checksum itself. The CRC chunk is always the last chunk appended to the frame.

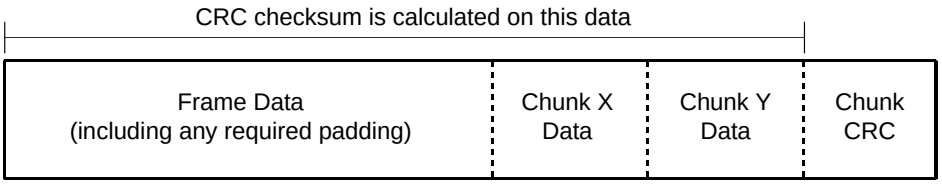



Fig. 46: CRC Checksum

| | |
|---|---|
| | The chunk mode must be made active before you can enable the time stamp feature or any of the other chunk features. Making the chunk mode inactive disables all chunk features. |

### Enabling the CRC Checksum and Retrieving Chunk Data

To enable the CRC checksum chunk:

- Use the Chunk Selector to select the CRC chunk.
- Use the Chunk Enable parameter to set the value of the chunk to true.

Once the CRC chunk is enabled, the camera will add a CRC chunk to each acquired frame.

To retrieve CRC information from a chunk appended to a frame that has been received by your PC, you must first run the frame and its appended chunks through the chunk parser included in the pylon API. Once the chunk parser has been used, you can retrieve the CRC information. Note that the CRC information provided by the chunk parser is not the CRC checksum itself. Rather it is a true/false result. When the frame and the appended chunks pass through the parser, the parser calculates a CRC checksum based on the received frame and chunk information. It then compares the calculated CRC checksum with the CRC checksum contained in the CRC checksum chunk. If the two match, the result will indicate that the frame data is OK. If the two do not match, the result will indicate that the frame is corrupted.

You can set the Chunk Selector and Chunk Enable parameter value from within your application software by using the pylon API. You can also run the parser and retrieve the chunk data. The following code snippets illustrate using the API to activate the chunk mode, enable the CRC checksum chunk, run the parser, and retrieve the frame counter chunk data:

```
// Make chunk mode active and enable CRC chunk
Camera.ChunkModeActive.SetValue( true );
```

```
Camera.ChunkSelector.SetValue( ChunkSelector_PayloadCRC16 );
Camera.ChunkEnable.SetValue( true );

// Check the CRC checksum of an acquired frame
IChunkParser &ChunkParser =
   *Camera.CreateChunkParser();
GrabResult Result;
StreamGrabber.RetrieveResult( Result );
ChunkParser.AttachBuffer( (unsigned char*) Result.Buffer(),
   Result.GetPayloadSize() );
if ( ChunkParser.HasCRC() && ! ChunkParser.CheckCRC() )
   cerr << "Image corrupted!" << endl;
```

For detailed information about using the pylon API, refer to the Basler pylon Programmer's Guide and API Reference.

You can also use the Basler pylon Viewer application to easily set the parameters.

For more information about the pylon Viewer, see Section 3 on page 17.

# 12 Troubleshooting and Support

This chapter outlines the resources available to you if you need help working with your camera.

## 12.1 Tech Support Resources

If you need advice about your camera or if you need assistance troubleshooting a problem with your camera, you can contact the Basler technical support team for your area. Basler technical support contact information is located in the front pages of this manual.

You will also find helpful information such as frequently asked questions, downloads, and application notes in the Downloads and the Support sections of our website: www.baslerweb.com

If you do decide to contact Basler technical support, please take a look at the form that appears on the last two pages of this section before you call. Filling out this form will help make sure that you have all of the information the Basler technical support team needs to help you with your problem.

## 12.2 Obtaining an RMA Number

Whenever you want to return material to Basler, you must request a Return Material Authorization (RMA) number before sending it back. The RMA number **must** be stated in your delivery documents when you ship your material to us! Please be aware that if you return material without an RMA number, we reserve the right to reject the material.

You can find detailed information about how to obtain an RMA number in the Support section of our website: www.baslerweb.com

# 12.3 Before Contacting Basler Technical Support

To help you as quickly and efficiently as possible when you have a problem with a Basler camera, it is important that you collect several pieces of information before you contact Basler technical support.

Copy the form that appears on the next two pages, fill it out, and fax the pages to your local dealer or to your nearest Basler support center. Or, you can send an e-mail listing the requested pieces of information and with the requested files attached. Basler technical support contact information is shown in the title section of this manual.

1 The camera's product ID: _____

2 The camera's serial number: _____

3 Network adapter that you use with the camera: _____

_____

4 Describe the problem in as much detail as possible: _____

_____

(If you need more space, use an extra sheet of paper.) _____

5 If known, what's the cause of the problem? _____

_____

_____

6 When did the problem occur?

☐ After start. ☐ While running.

☐ After a certain action (e.g., a change of parameters):

_____

_____

_____

_____

7 How often did/does the problem occur?

- ☐ Once.
- ☐ Every time.
- ☐ Regularly when: _____
- ☐ Occasionally when: _____

8 How severe is the problem?

- ☐ Camera can still be used.
- ☐ Camera can be used after I take this action: _____
- ☐ Camera can no longer be used.

9 Did your application ever run without problems?

- ☐ Yes
- ☐ No

10 Parameter set

It is very important for Basler technical support to get a copy of the exact camera parameters that you were using when the problem occurred.

To make note of the parameters, use Basler's pylon Viewer tool.

If you cannot access the camera, please try to state the following parameter settings:

- ☐ Frame Size: _____
- ☐ Pixel Format: _____
- ☐ Packet Size: _____
- ☐ Exposure Time: _____
- ☐ Line rate: _____

11 Live image/test image

If you are having an image problem, try to generate and save live images that show the problem. Also generate and save test images. Please save the images in BMP format, zip them, and send them to Basler technical support.

# Revision History

| Doc. ID Number | Date | Changes |
|---|---|---|
| AW00118301000 | 20 June 2012 | Preliminary release of this document. Applies to prototype cameras only. |

# Index

## A

acquisition frame count parameter ..... 71
acquisition start overtrigger event ..... 141
acquisition start trigger ..... 68, 69
acquisition status indicator ..... 115
acquisition status parameter ..... 115
acquisition trigger wait signal ..... 116
analog gain ..... 136
API ..... 16

## B

bandwidth assigned parameter ..... 30
bandwidth, managing ..... 31
binning ..... 147
bit depth ..... 2
black level
    mono cameras ..... 138
block diagram ..... 38
bus ..... 49

## C

cables
    Ethernet ..... 46
    I/O ..... 46
    power ..... 45
camera events ..... 143
camera power requirements ..... 47
chunk dynamic range max parameter ..... 168
chunk dynamic range min parameter ..... 168
chunk enable parameter
..... 170, 173, 176, 178, 180, 182
chunk encoder counter parameter ..... 178
chunk features, explained ..... 167
chunk frame counter parameter ..... 170
chunk frame trigger counter parameter ..... 177
chunk frame trigger ignored counter parameter ..... 177
chunk frames per trigger counter parameter ..... 177
chunk height parameter ..... 168
chunk input status at line trigger parameter ..... 181
chunk line trigger end to end counter parameter ..... 177
chunk line trigger ignored counter parameter ..... 177
chunk mode ..... 168
chunk mode active parameter ..... 168
chunk offset x parameter ..... 168
chunk parser .....
..... 168, 170, 173, 177, 178, 181, 182
chunk pixel format parameter ..... 168
chunk selector ..... 176, 178, 180
chunk time stamp parameter ..... 173
chunk width parameter ..... 168
cleaning the camera and sensor ..... 11
C-mount adapter ..... 6
code snippets, proper use ..... 11
configuration set loaded at startup ..... 165
configuration sets ..... 163–165
conformity ..... 2
connector types ..... 44
connectors ..... 39
CPU interrupts ..... 32
CRC checksum ..... 182

## D

debouncer ..... 52
default shading set file ..... 150
default startup set ..... 165
device firmware version parameter ..... 160
device ID parameter ..... 160
device manufacturer info parameter ..... 160
device model name parameter ..... 160
device scan type parameter ..... 160
device user ID parameter ..... 160
device vendor name parameter ..... 160
device version parameter ..... 160
digital gain ..... 136
dimensions ..... 2, 4
drivers, network ..... 17
DSNU
    see offset shading correction
dust ..... 10

## E

electromagnetic interference ..... 8
electrostatic discharge ..... 8
EMI ..... 8
enable resend parameter ..... 18, 20
encoder counter chunk ..... 178
environmental requirements ..... 9
ESD ..... 8
event overrun event ..... 141
event reporting ..... 141
exposure active signal ..... 114
exposure start delay ..... 81
exposure time
    maximum ..... 83
    minimum ..... 83
    setting ..... 84
exposure time abs parameter ..... 85
exposure time control modes
    off ..... 80
    timed ..... 80
    trigger width ..... 79
exposure time control off mode ..... 80
exposure time parameters ..... 84
exposure time raw parameter ..... 84
extended frame data ..... 168

## F

filter driver ..... 17
frame counter ..... 170
frame counter chunk
    reset ..... 171
frame retention parameter ..... 18
frame size ..... 65
frame start overtrigger event ..... 141
frame start trigger ..... 68, 72
frame start trigger activation parameter ... 73
    falling edge ..... 73
    level high ..... 73
    level low ..... 73
    rising edge ..... 73
frame start trigger mode parameter ... 72, 73
frame start trigger source parameter ..... 73
frame timeout ..... 76
frame timeout event ..... 76, 141
frame transmission delay parameter ..... 30
frame trigger counter ..... 175
frame trigger ignored counter ..... 175
frame trigger wait signal ..... 116
frames per trigger counter ..... 175
free run ..... 86, 88
frequency converter ..... 112
functional description ..... 37

## G

gain
    analog ..... 136
    digital ..... 136
    mono cameras ..... 135
gain shading correction ..... 149
gamma correction ..... 148

## H

heartbeat timeout parameter ..... 27
heartbeat timer ..... 27
heat dissipation ..... 9
height parameter ..... 65
horizontal binning ..... 147
humidity ..... 9

## I

I/O line response time ..... 63
input lines
    checking the state ..... 62, 180
    debouncer ..... 52
    electrical characteristics ..... 48
    inverter ..... 53
    termination resistor ..... 49, 52
input status at line trigger chunk ..... 180
installation
    hardware ..... 13
    software ..... 13
interface circuit ..... 49, 55
inter-packet delay ..... 18, 23, 32
inverter
    input lines ..... 53
    output lines ..... 58
IP configuration tool ..... 15
IP30 ..... 4

## J

jumbo frames....33
jumbo packets....33

## L

LEDs....39, 44
lens adapters....2
line inverter parameter....53, 58
line rate, max allowed....119
line source parameter....59
line start overtrigger event....141
line start trigger....68, 77
line status parameter....62, 63
line trigger end to end counter....175
line trigger ignored counter....175
line trigger wait signal....117
LUT (luminance lookup table)....144
LUT enable parameter....146
LUT index parameter....146
LVTTL....51

## M

max height parameter....160
max number resend request parameter....23
max width parameter....160
maximum exposure time....83
maximum line rate....119
minimum exposure time....83
minimum line rate....2
minimum output pulse width....58, 117
missing packet
  detection....19
  status....19
models....1
mono 12 packed pixel data format....127
mono 12 pixel data format....125
mono 8 pixel data format....124
mounting holes....4
multiple cameras on a network....31

## N

network drivers....17
network parameter....32
network performance....32

## O

offset shading correction....149
output lines
  checking the state....62
  electrical characteristics....55
  inverter....58
  minimum output pulse width....58
  setting the state....60
  user settable....59, 60

## P

packet size parameter....29
packet timeout parameter....18, 23
parameter limits, removing....140
parameter sets....163
parameter sets, saving....164
parameters loaded at startup....165
partial closing frame parameter....73
payload size parameter....29
performance driver....17
pin assignments....41, 42
pin numbering....40
pixel data formats....123
  mono 12....125
  mono 12 packed....127
  mono 8....124
  YUV 422 (YUYV) packed....131
  YUV 422 packed....129
pixel format parameter....123
pixel size....2
pixel transmission sequence....133
power cable....45
  voltage requirements....47
power requirements....2
PRNU
  see gain shading correction
protection class....4
pylon API....16
pylon Viewer....15

## R

read timeout parameter....27
receive descriptors....32
receive window....19
receive window size parameter....20
remove limits parameter....140

removing parameter limits ..... 140
resend request batching parameter ..... 21
resend request response timeout parameter ..... 23
resend request threshold parameter ..... 21
resend timeout parameter ..... 23
response time, I/O lines ..... 63
return material authorization ..... 185
RMA number ..... 185
RS-422 ..... 48
  bus ..... 49
RS-644 LVDS ..... 50

## S

saving parameter sets ..... 163, 164
sensor
  architecture ..... 38
  pixel size ..... 2
  size ..... 2
  type ..... 2
sensor board temperature parameter ..... 160
sensor height parameter ..... 160
sensor width parameter ..... 160
serial number ..... 11
sets of parameters, saving ..... 164
shading correction ..... 149
  gain ..... 149
  offset ..... 149
shading file ..... 149
shading set ..... 149
shading status ..... 153
shaft encoder module counter mode parameter ..... 105
shaft encoder module counter parameter ..... 105
shaft encoder module max parameter ..... 105
shaft encoder module mode parameter ..... 105
shaft encoder module reset command ..... 105
shaft encoder module reverse counter max parameter ..... 105
shaft encoder module reverse counter reset command ..... 105
shaft encoder software module ..... 104
software development kit ..... 16
speed and duplex ..... 32
startup parameter set ..... 165
support ..... 186

## T

technical support ..... 185
temperature, housing ..... 9
termination resistor ..... 49, 52
test images ..... 157
time stamp ..... 173
timed exposure time control mode ..... 80
transition threshold ..... 51
trigger
  acquisition start ..... 68
  frame start ..... 68, 72
  line start ..... 68, 77
trigger counters ..... 175
trigger delay ..... 154
trigger width exposure time control mode 79

## U

use case
  description ..... 86
  diagram ..... 86
user defined values ..... 162
user output value parameter ..... 60
user settable output lines ..... 59, 60
user shading set file ..... 150

## V

ventilation ..... 9
viewer ..... 15
voltage requirements
  LVTTL ..... 51

## W

weight ..... 2
width parameter ..... 65
write timeout parameter ..... 27

## X

x offset parameter ..... 65

## Y

YUV 422 (YUYV) packed pixel data format..
.....131
YUV 422 packed pixel data format.....129